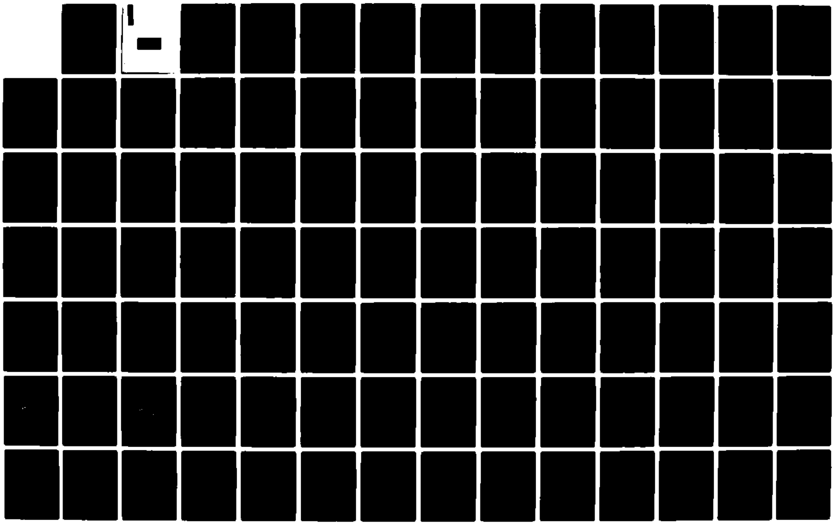
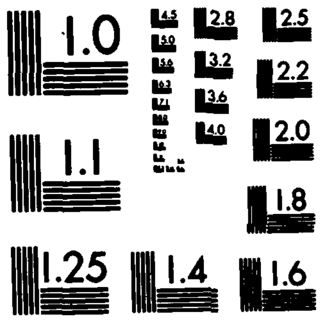
AD-A124 893 IMPROVING THE USEABILITY OF THE DEFENSE MAPPING AGENCY DIGITAL FEATURE AN..(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. M R NICOL

UNCLASSIFIED SEP 82 AFIT/GCS/EE/82S-9 F/G 9/2 NL

1/3





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AFIT/GCS/EE/82S-9

(1)

IMPROVING THE USEABILITY OF THE DEFENSE MAPPING AGENCY DIGITAL FEATURE ANALYSIS DATA FOR TRAINING SIMULATOR APPLICATIONS

THESIS

AFIT/GCS/EE/82S-9 Michael R. Nicol

Approved for public release; distribution unlimited

AFIT/GCS/EE/82S-9

IMPROVING THE USEABILITY OF THE DEFENSE MAPPING AGENCY DIGITAL FEATURE ANALYSIS DATA FOR TRAINING SIMULATOR APPLICATIONS

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

by

Michael R. Nicol

September 1982

Preface

This paper documents the design and development of a software package that is used to detect and correct certain types of anomalies or errors that are present in the digital cartographic data used as source information in the generation of various types of simulator data bases. The paper is directed primarily to those persons who are familiar with the Defense Mapping Agency's Digital Feature Analysis Data (DFAD), and to those who will be using this software to improve the quality of the data. However, I hope that it will also provide helpful information to anyone who is interested in simulator or cartographic data bases, or who is considering the use of DFAD for some other application.

I wish to express my appreciation to a number of people who have been instrumental in helping or allowing me to pursue and complete this thesis project. First, thanks to Richard O'Dell of Aeronautical Systems Division Flight Simulator Branch, and Robert Beck from ASD's Deputy for Simulators, Engineering Directorate, for their confidence and support in allowing me to initially attempt to solve this problem. Thanks to the staff at the Naval Training Equipment Center, Advanced Computer Systems Laboratory, Orlando, Florida, for their support and the use of their Vax 11/780 computer system, where the basic concepts of this software package were first developed. Thanks also to my Branch Chief, Major Stan Owen, for his patience during the development of the software, and to the staff of the ASD/ENET Digital Imaging Facility, where the software was refined into a useable system. Additionally, I would like to thank my Advisor, Lt Col Alan Ross, for the guidance, reassurance, and understanding he provided throughout this effort. Thanks also to Dr. Thomas Hartrum and Professor Charles Richard for their review and critical support during the preparation of this document.

Above all, I wish to express my most sincere appreciation to my wife Cheryl, for her patience and understanding as this effort was conceived, designed, and implemented. Thanks are also due to her for her tireless dedication in typing this report.

Contents

List of Figures

List of Tables

Abstract

The Air Force is intensifying its reliance upon flight simulators to provide initial and proficiency training for various aircrew members. The effectiveness of these simulators depends heavily upon the Defense Mapping Agency's digital cartographic data, which is used as source data to generate simulator data bases for visual, radar, and sensor simulations. Unfortunately, the source data occasionally displays certain characteristics which disrupt the highly automated processes used to generate these simulator data bases, or which lead to improper or visually distracting effects in the resulting simulations.

This paper describes the development of a software package designed to detect these problems in the source data. The capability is also provided to interactively, or in some cases automatically, correct the problems, thus producing slightly modified data that is free of these disruptive characteristics. This software streamlines the simulator data base generation process and ultimately results in a higher fidelity simulator data base.

IMPROVING THE USEABILITY OF THE DEFENSE MAPPING AGENCY DIGITAL FEATURE ANALYSIS DATA FOR TRAINING SIMULATOR APPLICATIONS

I. Introduction

Background

The Air Force has historically relied upon experience gained from hours in the actual aircraft as a primary source of aircrew training. However, with the advent of the energy crisis and the associated rising costs of fuel in the past decade, the Air Force has focused increased attention on the use of training simulators to provide full mission training for various aircrew members. Such full mission simulators, known as weapon system trainers, often simulate a full range of aircraft systems and provide a number of appropriate sensory responses to the trainee. For example, the cockpit instruments, sensor and radar displays, and out-the-window visual scenes may all be simulated in a real-time manner, such that responses to crew member inputs occur at approximately the same speed as in the actual aircraft. Thus a pilot can actually "fly" the simulator, and a radar navigator can trace the course of the "aircraft" as it moves through the simulated world.

Each simulator visual, radar, and sensor system requires a data base, which is a computerized model of the real world that defines the simulator environment or gaming area. Figure 1 shows a typical simulator visual system configuration. The on-line data base is accessed to

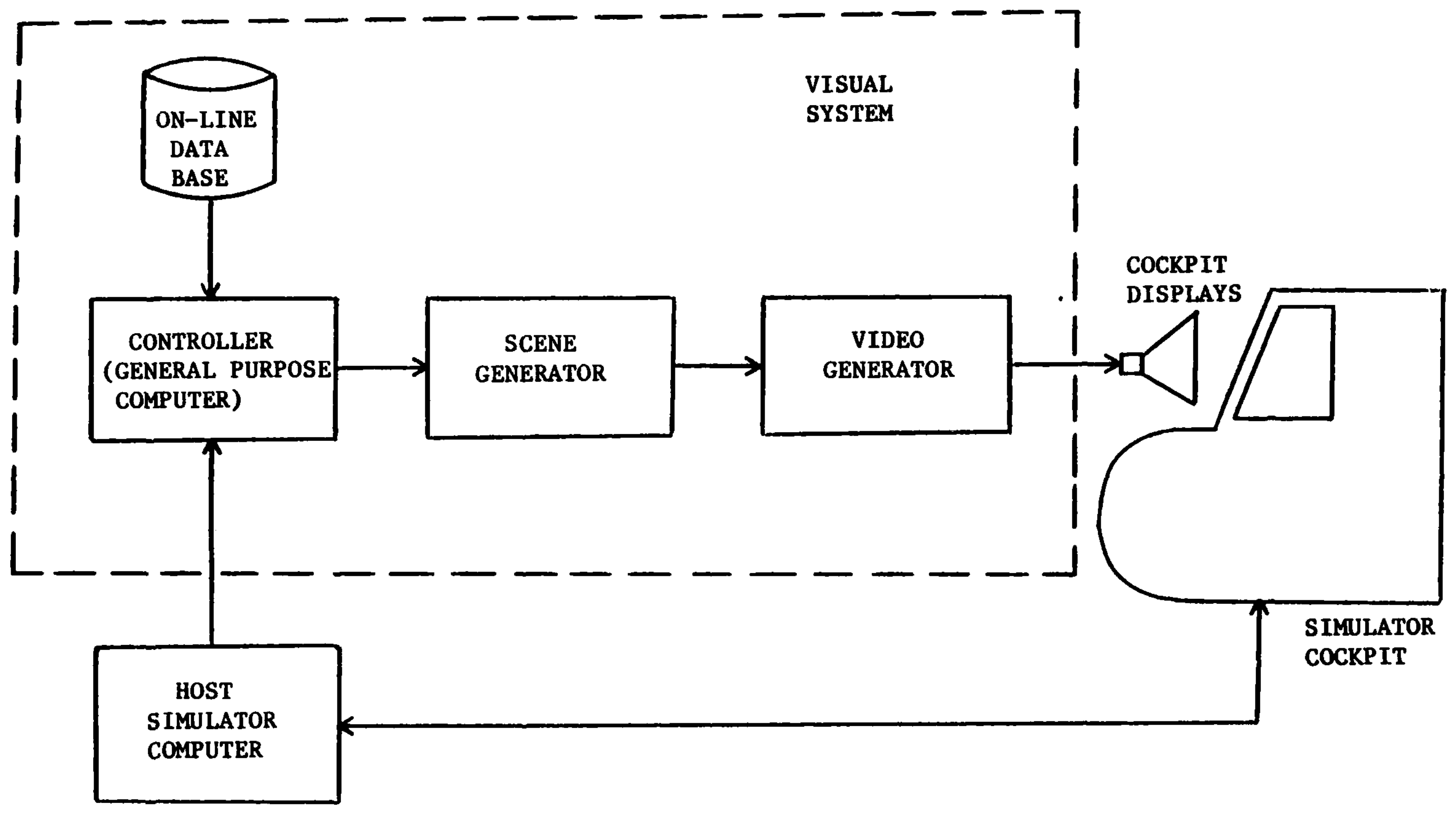


Fig 1. Simulator Visual System

provide the data from which the imagery for display at the cockpit windows is generated. The host simulator computer provides aircraft location and attitude data which is used to decide which part of the data base to access, and to calculate the appropriate viewing parameters to be used to generate the next image. This entire process produces a new image, or frame, at a rate of 30 per second.

Often these on-line data bases may cover geographic areas of thousands of square nautical miles. The most commonly available source data used to generate these on-line data bases is produced by the Defense Mapping Agency (DMA). The visual, radar, or sensor data base is created by a computerized process called transformation, in which a complex computer program, called a digital data base transformation program (DDBTP), operates on the DMA source data to produce a simulator data base in a format that is compatible with the simulator image generation system.

DMA produces two types of data which are commonly used in the generation of simulator data bases (Ref 2). The Digital Terrain Elevation Data (DTED) consists of one degree square arrays of terrain elevation information provided at a three arc-second spacing in latitude and longitude. The transformation program typically fits a set of surfaces to this elevation data to form the basic shape of the earth's surface for the simulator data base. The other type of DMA data is Digital Feature Analysis Data (DFAD), also known as culture data, which describes natural and man-made features which lie on top of the terrain. Examples of features described by the DFAD include forests, lakes, cities, airfields, power plants, factories, power transmission lines, and numerous others. This information is merged with the terrain elevation data to form a complete simulator data base.

The DFAD data is provided as a collection of manuscripts on a magnetic computer tape. A manuscript typically covers a geographic area of one-fourth to a full square degree in latitude/longitude. Manuscripts are mosaicked together to provide continuous geographic coverage. An example of manuscript coverage of a geographic area is depicted in Figure 2. A manuscript header record provided with each manuscript contains information on the content of the manuscript, its size, and its location.

Each manuscript contains a number of features, each of which is classified as one of three basic types. Point features are used to portray discrete features such as microwave towers, smokestacks, or buildings, which can adequately be defined by length, width, height, and center location. Point features may also be used to describe powerlines, and multiple center point locations are provided to locate the individual powerline pylons. Linear features portray those real-world features which exhibit an essentially standard width, such as runways, causeways, bridges, or treelines. Areal features are used to portray real-world features such as forests, lakes, or housing areas, where the feature definition bounds an essentially homogeneous area. Examples of each type of feature are shown in Figure 2. Each feature has header information which includes a feature analysis code (FAC), feature identification code (FIC), feature type (point, linear, or areal), predominant height, surface material category, the number of points which define the feature location or boundary, and several parameters which are

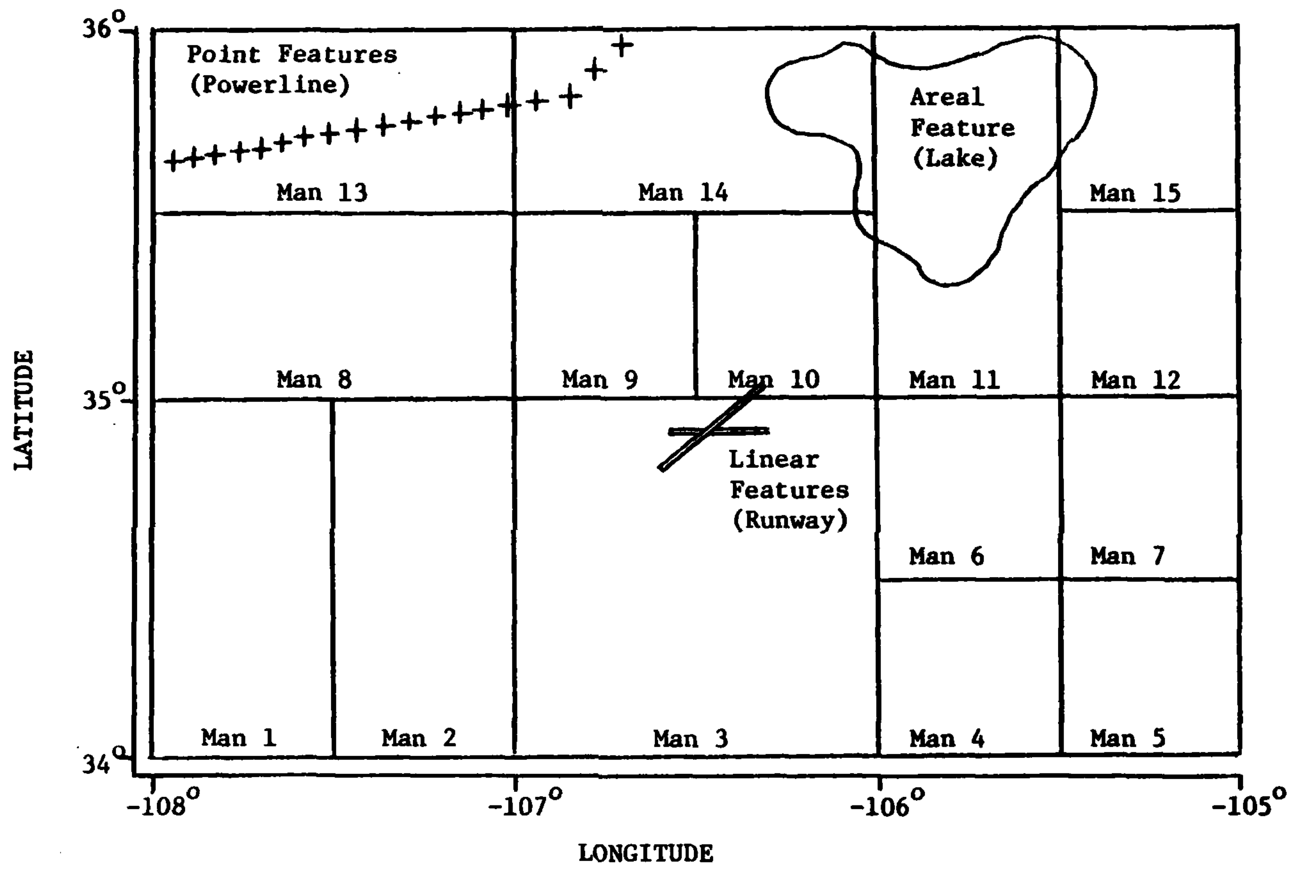


Fig 2. DFAD Manuscripts and Features

peculiar to the feature type. These header parameters are discussed briefly below.

The FAC number is a code which specifies a feature's relative priority within a manuscript. The first feature on a manuscript is FAC number one, and it is a special case areal feature that delimits the manuscript area of coverage. All successive features have higher FAC numbers, and take precedence over, or lie on top of, any features with a lower FAC number. Thus a soil feature with a high FAC can portray a clearing when it is positioned to geographically overlay part of a surrounding forest which has a lower FAC number. The FIC number is one of 264 defined codes which specify whether a feature is a school, lake, dam, observatory, rice paddy, or some other type of feature. A list of the defined FIC codes is provided in Appendix B. The predominant height is the average height of the feature. The surface material category classifies the feature surface composition as wood, asphalt, metal, water, or another of thirteen codes. Table I lists the defined surface material categories.

Each feature also contains a list of one or more points, or coordinates, which are referenced to the southwest corner, or origin, of the manuscript, and which define the location, path, or boundary of a point, linear, or areal feature, respectively. These points are also referred to as delta sets, since they express a delta location relative to the manuscript origin. These points have a resolution of one-tenth arc-second in latitude and longitude.

With this basic understanding of the DMA culture data (DFAD), a discussion of the problem to be addressed is possible. As technology

TABLE I

DFAD Surface Material Categories

Code	Description
****	***********
1	Metal
2	Part Metal
3	Stone/Brick
4	Composition
5	Earthen Works
6	Water
7	Desert/Sand
8	Rock
9	Asphalt/Concrete
10	Soil
11	Marsh
12	Trees
13	Snow/Ice

has advanced and the capacity of visual and other simulator image generation systems has increased, the transformation programs which generate the data bases for these systems have become necessarily more complex, and the data bases have become more sophisticated. Thus, a higher percentage of features from the DFAD are being processed and included in the resulting simulator data base. However, occasional anomalies or errors in the DFAD portrayal of some features can cause catastrophic failures of the transformation program, or can allow erroneous or visually distracting information to be passed through the transformation process. One of the primary problems involves self-intersecting features. Such features are characterized by intersections between the line segments which define the path of a linear feature, or the boundary of an areal feature. Examples of these self-intersecting features are provided in Figure 3. Such features cannot be processed properly by

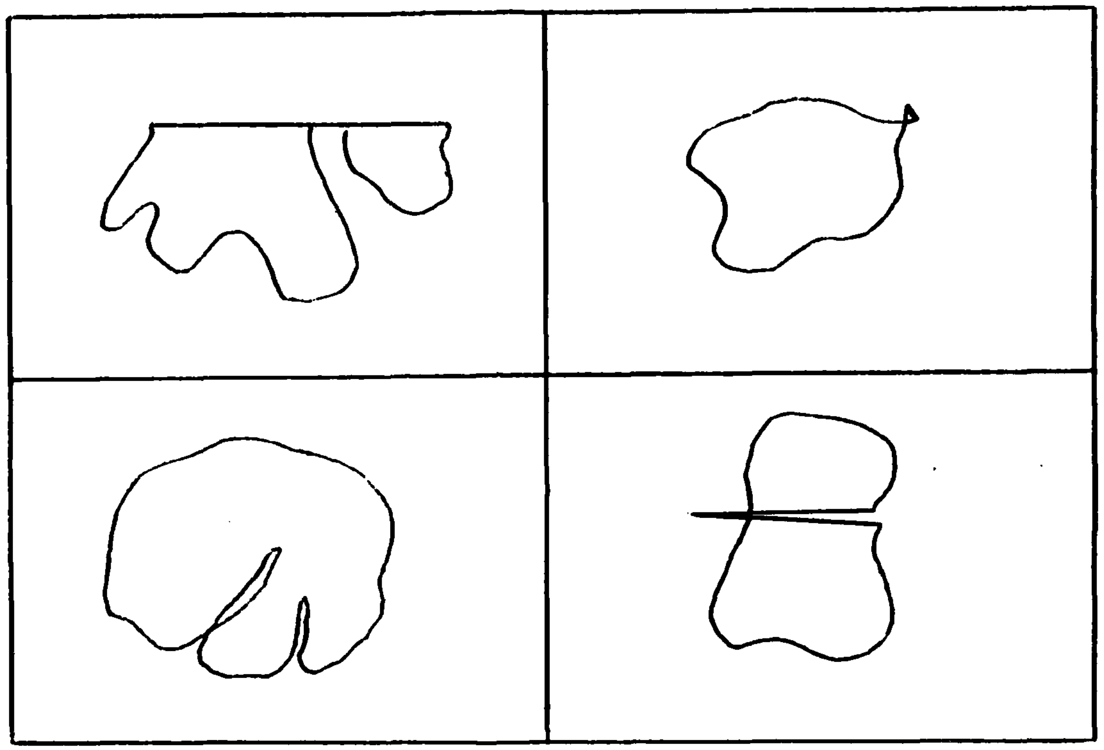


Fig 3. Self Intersecting Features

the transformation program. Furthermore, DMA has no capability to detect or eliminate the intersections. In addition, a number of other errors exist which also cause problems for transformation programs.

The Air Force thus has two options. The first is to ignore or leave out the features with errors. This approach reduces the content and degrades the fidelity of the resulting simulator data base. The alternative is to scan all of the DFAD source to detect the errors and eliminate them prior to the transformation process. The first option is unacceptable, since the large, complex features which contribute important information to the data base are the ones which often contain errors. Conversely, no capability exists to detect and correct the errors. This situation leads to the following problem statement.

Statement of the Problem

The Defense Mapping Agency Digital Feature Analysis Data contains a number of anomalies, or errors, which are randomly scattered throughout the data. DMA has no ability to detect or correct these errors, and their presence causes severe problems during transformation or during the use of the resulting data base. This problem is common to most current visual, radar, and sensor simulation transformation processes. If a transformation program detects such an error in a critical feature, that feature is usually eliminated from further processing. Dealing with these errors adds a difficult and unpredictable step to an already time-consuming and expensive process. In addition, the same errors may be encountered and handled repeatedly as individual users of the DFAD transform the same geographic areas for different applications.

Proposed Solution

The obvious solution to this problem is to devise a capability to detect and eliminate the errors at the source, which in this case is the DFAD digital file. This is the approach taken in this effort. The culture data, once placed in DMA's cartographic data base, is difficult to modify, since DMA's primary computer system is already overloaded and is used almost exclusively in a batch environment. Thus to change even a single feature parameter first requires knowledge of the existence of a problem, followed by development and submission of a card deck for batch processing. This causes further delays because of slow turnaround due to the overloading of the system. Once the changes are implemented, further analysis must be performed to insure that the change was properly inserted and did not introduce additional problems.

The solution to this problem involves the design, development, and implementation of a software package with the capability to detect the errors and interactively correct them. In addition, the feasibility of automated correction techniques is addressed for certain types of errors. To avoid the batch environment and turnaround limitations, this software package can be implemented on a VAX 11/780 computer system which is available at DMA. Once the software is implemented, DMA can perform the error correction, and users of the data will no longer need to devote time and effort to this problem.

The purpose of this project, then, is to design, develop, and implement a software package that can be used to correct, automatically if possible, these troublesome errors in the DFAD. This proposed software package is collectively referred to as DFADEDIT.

Approach

The approach employed to develop the DFADEDIT software involves four basic steps. The first is a system requirements definition in which the errors to be addressed, and the means of addressing them, are defined. The results of this step are documented in Chapter II of this report. The second step consists of the establishment of a software structure and data flow which support the detection and correction of these errors. This step is discussed in Chapter III.

The third step involves the detailed design and development of algorithms to detect and correct the subject errors. Algorithm development is the topic of Chapter IV. The fourth and final step is the implementation and test of the resulting DFADEDIT software on the VAX 11/780 computer system. Efforts leading to the achievement of this step are documented in Chapter V.

Chapter VI summarizes the conclusions drawn from this effort, and provides a number of recommendations for further consideration and possible future implementation. Several appendices which contain supporting information are also provided. Of particular interest is the DFADEDIT User's Guide provided in Appendix A.

II. System Requirements Definition

This chapter has a twofold purpose. The first is to define and describe the specific anomalies or errors that are to be addressed by this project. The second is to discuss various implications and considerations that arise in the detection and correction of these errors. The feasibility of reliable automated correction for certain types of errors is also addressed.

The development of these requirements is based primarily on experience from within the Air Force Aeronautical Systems Division (ASD) Deputy for Simulators. Most of the errors to be addressed by the software have been encountered by ASD or contractor personnel when attempting to use the DFAD. Interaction with DMA personnel has led to further refinement of system requirements.

Errors Addressed

The DFAD contains various types of errors, some of which are difficult to properly correct without access to the source data from which the DFAD was derived. Conversely, a number of the error types encountered can be corrected without access to the source data, using techniques which result in only minor changes to the feature definition. Fortunately, the latter case is the most frequent for the types of errors which affect simulator applications, and therefore many of the features with errors can be corrected very quickly once the errors have been identified by the software. The types of errors which qualify for this approach are discussed below.

Although errors in the manuscript header data are extremely rare, all manuscript header parameters must be tested for validity. Manuscript header parameters will be checked against valid ranges or values defined in the specification (Ref 2). In most cases the specification defines a finite number of valid values, or a bounded range of values. Should an illegal value be detected for any parameter, the DFADEDIT software will issue an error message. In other cases, a parameter may not have a finite range, but the actual parameter value may exceed what would be considered a logical value. In this case, the DFADEDIT software will issue a warning message so that the user can examine the parameter in question and make a decision as to its validity. In either case, the software will provide the user with a means to access and modify the appropriate manuscript header parameter.

As discussed previously, the first feature on every manuscript is feature analysis code (FAC) one. This feature must always be an areal feature with four points which define the corners of the manuscript. The specification allows for some non-rectangularity in the manuscript boundary. However, the software will flag any non-rectangularity as an error due to its potential to introduce gaps or overlaps relative to adjacent manuscripts, and the associated difficulty in aligning real-world features which cross these skewed manuscript boundaries. The specification also constrains the surface material category for FAC 1 features to be water or soil. If any of these conditions for the FAC 1 feature are not met, the software will issue an error message and will provide the user with the ability to access and modify the FAC 1 feature header or coordinate data.

The software must check the feature header parameters as each feature on the manuscript is processed. The FAC number must increase with each new feature. Feature type, feature identification code (FIC), surface material category, predominant height, and number of feature points all have defined valid ranges and must be tested by the software. Several feature type-dependent header parameters must also be checked against their valid ranges. Again, if any errors are found, the software will provide a means to resolve them.

Linear features must be checked to determine if they close upon themselves. A linear feature is "closed" if the last point defining the feature is geographically identical to the first. A closed linear feature is not necessarily incorrect, but the software will warn the user of such an occurrence in case it is actually a mislabeled areal feature.

Areal features, conversely, should be closed. The boundary of an areal feature as defined by the feature points encloses a homogeneous area, therefore the last point of an areal feature must always be geographically identical to the first. The software will always test for areal feature closure, and will attempt to close any open areal feature. All occurrences of open areal features will be documented.

Areal features must also be examined for the potential occurrence of "wrap-around feature" errors. This error occurs when a large feature with a relatively high FAC number mistakenly overlays a number of features with lower FAC numbers, due to improper digitization or coding during the production process. This error has the effect of covering up, or eliminating, underlying features with lower FAC numbers. The capability to interactively correct wrap-around feature errors will also be provided by the DFADEDIT software.

Areal features must be tested for zero area and for clockwise digitization. An areal feature whose points are all colinear encloses zero area and thus is not a legal areal feature. The specification also states that all areal features will be digitized in a counter-clockwise direction, such that if the boundary is traversed in the direction of digitization, the interior of the feature will always be to the left. Occasionally, a feature or part of a feature is digitized in a clockwise direction. The software will provide a method to interactively correct such errors.

All feature coordinates must be tested for the occurrence of points which lie outside the manuscript boundary as defined by the line segments which join the coordinates of the FAC 1 feature. All feature points must also be checked for the existence of stack and spike points. Stack points occur when two consecutive points are geographically equal. Spike points occur when the first and third points of a set of three consecutive points are geographically equal. The correction of stack and spike points is straightforward, and will be implemented as an automated function. The correction of points outside the manuscript is a less exact process, since the proper corrective action varies from case to case. Thus, this type of error will be corrected interactively.

The DFADEDIT software will check all linear and areal features for self-intersecting line segments. An intersection occurs when any two non-consecutive line segments of a feature touch or cross, or when any two consecutive line segments touch at some point other than their common endpoint. This is one of the most frequent and troublesome errors, and it is also the most computationally complex error to detect. The test to detect intersections is thus relatively time consuming. Correction of intersection errors for some features can also be very tedious, and for this reason, automated correction of intersections will be investigated.

All linear and areal features must also be checked for the occurrence of exit point errors. The specification states that when a feature exits the manuscript, the point where the feature exits will be flagged with a negative latitude. An exit point error occurs when a valid exit point is not so flagged, or when a point which is not a geographically legal exit point is flagged. Exit point errors are also relatively common, and the required corrective action is generally straightforward. Thus, automated correction of exit point errors will also be investigated.

Finally, if the containing manuscript as defined by the FAC 1 feature is not rectangular (is skewed), each linear and areal feature will be tested to see if potential exit point errors exist. These potential exit point errors may become exit point errors once the manuscript boundary skew is corrected. The software will warn the user of all potential exit point errors.

Each of the error conditions described above are deficiencies whose existence in the DFAD has been demonstrated. The DFADEDIT software will have the capability to detect all instances of such errors, and will issue appropriate error or warning messages. Automated correction is desirable if it can be performed in a reliable manner, but at common endpoint. This is one of the most frequent and troublesome errors, and it is also the most computationally complex error to detect. The test to detect intersections is thus relatively time consuming. Correction of intersection errors for some features can also be very tedious, and for this reason, automated correction of intersections will be investigated.

All linear and areal features must also be checked for the occurrence of exit point errors. The specification states that when a feature exits the manuscript, the point where the feature exits will be flagged with a negative latitude. An exit point error occurs when a valid exit point is not so flagged, or when a point which is not a geographically legal exit point is flagged. Exit point errors are also relatively common, and the required corrective action is generally straightforward. Thus, automated correction of exit point errors will also be investigated.

Finally, if the containing manuscript as defined by the FAC 1 feature is not rectangular (is skewed), each linear and areal feature will be tested to see if potential exit point errors exist. These potential exit point errors may become exit point errors once the manuscript boundary skew is corrected. The software will warn the user of all potential exit point errors.

Each of the error conditions described above are deficiencies whose existence in the DFAD has been demonstrated. The DFADEDIT software will have the capability to detect all instances of such errors, and will issue appropriate error or warning messages. Automated correction is desirable if it can be performed in a reliable manner, but at least automated detection with interactive correction capability, supported if necessary by manual analysis, will be provided for each type of error.

System Requirements

The DFAD digital data is provided to users via nine-track magnetic tape. The tape is thus a convenient input medium for the error correction software. The output of the error correction process will also be provided on a magnetic tape, formatted identically to the input tape, with all corrections implemented. The process of detecting and correcting errors is well suited to computer solution.

In determining the attributes of a computer system to support the error correction software, several considerations must be addressed. The error correction software involves several programs which require in excess of 100,000 32-bit words of memory for program and data storage. Thus the target computer system must provide the user access to a relatively large amount of memory. In addition, some of the algorithms involve highly repetitive computations. For example, the intersection detection process is very time-consuming, and thus the target computer system must be a state-of-the-art machine which possesses sufficient speed to process data at a reasonable rate. The system must also be capable of servicing multiple users concurrently to support the various stages of error correction activities in parallel.

Two nine-track tape drives are required to support the merging of interactively corrected data from disk files with an input DFAD tape to produce a corrected output tape. In addition, disk space must be available for several internal data files which are created and used during the error correction process.

Since interactive correction is often required, an interactive graphics capability is a must. A calligraphic graphics terminal is preferable because of the generally higher resolution available and because there is little justification for raster graphics. A graphics hard-copy capability would be useful, but is not an absolute requirement.

Finally, the needs of the production environment must be considered. Since Defense Mapping Agency is production oriented, and since contractual obligations are often involved, the DFADEDIT software must address two extremely important issues. The first is traceability. Any changes to a manuscript header or to any feature, whether done automatically or interactively, must be fully documented. Secondly, the software must be reliable. That is, error correction should be a one-pass process which prevents undetected errors from slipping through loopholes in the system. Special attention must be focused on automated corrections to insure that the software does not allow an automatically corrected feature that is algorithmically satisfactory, but unacceptable by normal cartographic standards, to pass through the error correction process undetected.

The software system proposed to satisfy all these criteria is the subject of the next chapter.

III. System Definition

This chapter describes the software system developed to satisfy the requirements outlined in the previous chapter. The system development is traced from initial concept development and demonstration through the definition of the final configuration implemented on the VAX 11/780 system at DMA.

Concept Demonstration

When ASD was first faced with implementing error correction procedures, self-intersecting features were the critical problem, and rapid turnaround was required due to Air Force obligations for the delivery of the DFAD to a contractor. A VAX 11/780 computer system that could support the requirements defined in the previous chapter was located at the Naval Training Equipment Center (NTEC) in Orlando, Florida. The system possessed the required disk storage and magnetic tape drives, and a Tektronix 4014 terminal was available to support the interactive graphics functions. The advanced memory management capabilities of the VAX made it an ideal machine for the large programs which had to be developed and implemented in a very short time. Thus, the initial concept demonstration version of this software, addressing primarily intersection detection and correction, was implemented in FORTRAN on the NTEC VAX.

Once the initial concept was demonstrated, DMA became interested in enhancing the error-correction capability and implementing the software on a VAX system at the DMA Aerospace Center in St. Louis, Missouri. A VAX system was available at the Air Force Institute of Technology

(AFIT), and the preliminary plan was to develop the software on the AFIT VAX through compilation and linking, and then test the software thoroughly at DMA. Complete test of the software at AFIT was impossible since no tape drives were available.

At about the same time, the Simulator Division Visual and Electro-Optical Branch (ASD/ENETV) of the ASD Directorate of Equipment Engineering was bringing the Digital Imaging Facility (DIFAC) on-line. Although the general purpose computer in this facility was a Systems Engineering Laboratories (SEL) 32/7780, it had the required memory capacity, tape drives, disk storage, and Tektronix graphics terminal to support the software development and test. Since a long test cycle was envisioned due to the complexity of the software, and since the time to convert the software from SEL to VAX FORTRAN was predicted to be significantly less than that required to fully test the software, the development plan was modified to allow for development and test of the DFADEDIT software on the DIFAC computer. Once testing was completed, the software would be converted to a VAX implementation and transferred to the DMA Aerospace Center for final testing. This approach minimized the scope of testing on the VAX.

Target System Hardware

The DMA VAX system had been proven capable of supporting the computational requirements of the error correction process by virtue of the concept demonstration implementation at NTEC. The DMA system also had the two required nine-track magnetic tape drives along with four 300 megabyte disk drives, which were more than adequate for program and data storage.

One complication was introduced by the lack of any Tektronix graphics terminals on the DMA VAX. However, four Hewlett-Packard HP-2649C terminals are available on the system. These terminals have both a graphics and alphanumeric memory, and could thus simultaneously fulfill both the functions of the user interactive terminal and user graphics terminal. Although the useable graphics resolution of the HP-2649C for this application is limited to a 360 x 360 raster and no hard-copy capability exists, the terminal is generally adequate for the graphics requirements of the DFADEDIT software.

Thus it was determined that the existing in-house system at DMA was capable of supporting the error-correction software. The original concept demonstration system was implemented at DMA for interim use while the enhanced software was developed on the ASD DIFAC facility. All concerns as to the useability of the HP-2649C terminals were eliminated by the successful implementation of the concept demonstration software.

DFADEDIT Software Architecture

Once the hardware requirements had been addressed, the enhanced software architecture began to evolve. The general flow of the system involves processing features on a manuscript basis from an input DFAD tape, detecting all errors, and correcting them automatically or flagging them for interactive correction during initial processing. Once all corrections were complete and verified, the original and corrected features would be merged and a corrected output tape could be generated to be used as a replacement for the original tape. A diagram of the system designed to perform these functions is shown in Figure 4. The system consists of eleven programs and up to five types of internal or intermediate data storage files. Each program and file type is addressed briefly below. Further details are available in the DFADEDIT User's Guide, provided in Appendix A.

The DFADSTAT program provides statistics on the contents of a DFAD input tape. These statistics are useful in determining the number of manuscripts and their relative complexity or density, geographical areas of coverage for individual manuscripts or for the entire tape, and types of features contained on the tape. The statistics are also helpful in comparing the contents of a corrected tape to the corresponding original tape.

The DFADPLOT program is used for plotting the full contents or selected subsets of individual manuscripts on the user's graphics terminal. This program is not specifically required for error correction, but is provided as a general tool for inspection of the data.

The CHECK program is the workhorse of the DFADEDIT system and is the first step in the error correction process. It provides the full range of error tests, and is capable of automated correction of some types of errors. It produces a directory file which contains an entry for each feature with errors. Each feature with any type of error is assigned an error severity code which is included in the directory file entry for that feature. CHECK also produces a review file which contains both the original and corrected versions of features which are corrected automatically. CHECK produces an intermediate DFAD tape which

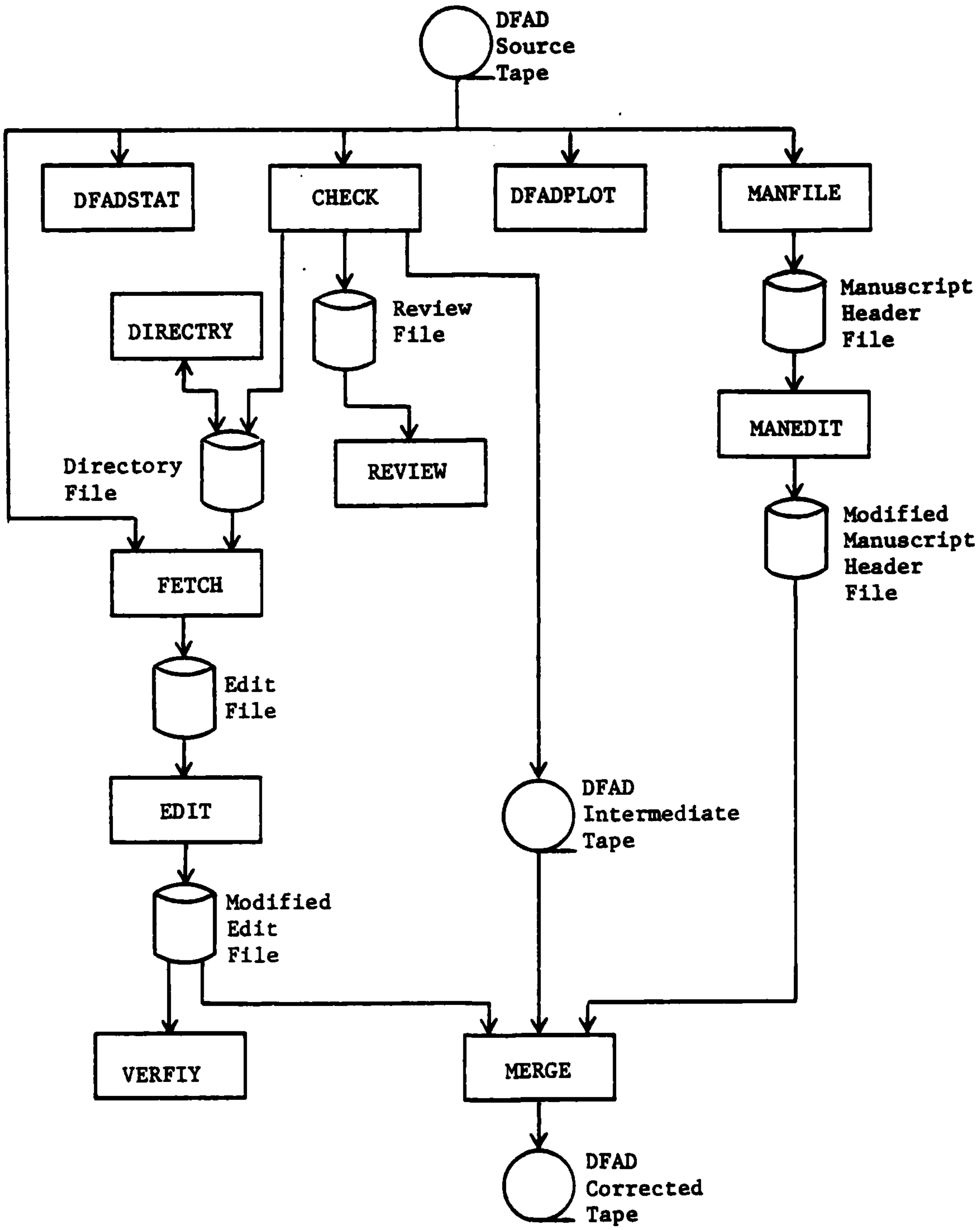


Fig 4. DFADEDIT Software Configuration

is identical to the input tape except that features which are successfully corrected automatically are substituted for the originals. CHECK documents all errors it discovers and all automated corrections it performs.

If CHECK discovers manuscript header errors, program MANFILE is used to read the DFAD input tape and generate a manuscript header file. This file contains an entry for each manuscript header on the input tape. Program MANEDIT is then used to interactively modify the parameters which are in error. MANEDIT produces a corrected manuscript header file and a listing which logs all user modifications. The corrected manuscript header file will be merged onto the corrected tape later in the error correction process. If no manuscript header errors or warnings are detected by CHECK, programs MANFILE and MANEDIT need not be used in the error correction process for a particular tape.

Program REVIEW is used to interactively examine the contents of the review file generated by CHECK. REVIEW allows the user to quickly determine the validity of automated corrections, using a number of graphical and analytical techniques which can be applied to both the original and corrected feature representations. The user controls the review process with 19 interactive commands. If the user determines that all automated corrections are acceptable, no further action need be taken, since the corrected features have already been substituted for the originals on the intermediate DFAD tape. However, if the user discovers an unacceptable automated correction, he must use the DIRECTRY program (discussed below) to modify the directory file so that the subject feature is placed in the edit file for interactive correction. This edit file feature, once corrected, will replace the unacceptable automatically corrected feature from the intermediate tape during MERGE processing (also discussed below).

Program DIRECTRY is an interactive program which allows the user to modify the contents of the directory file. Thus features which require modification for reasons other than errors detected by CHECK may be added to the directory file. If the user discovers unacceptable automated corrections using REVIEW, he must use DIRECTRY to change the directory file entry for that feature, in effect flagging the feature for interactive correction. The feature will then be included in the edit file generated by FETCH.

Program FETCH is used to create the edit file using the directory file and the original DFAD tape as input. As FETCH reads each record of the directory file, it examines the error severity assigned to the subject feature. If FETCH determines that the feature requires interactive correction, the feature is retrieved from the input DFAD tape and the entire feature definition is written to the output edit file.

Program EDIT provides the user with a powerful set of interactive analysis, plot, and modification capabilities. EDIT allows the user to sequence through the edit file feature by feature. A series of 40 interactive commands are available to plot features alone or relative to the containing manuscript, to check for various types of errors, to modify, move, delete, or insert points, and to change feature header parameters. Additionally, the user can delete entire features, create features from scratch, or insert features from other edit files. EDIT documents all changes implemented by the user and produces an updated edit file which contains the modified or corrected features.

Program VERIFY provides a means to check corrected edit file features for the presence of residual errors or new errors introduced by the user during the EDIT process. VERIFY is a batch program which can process a complete edit file or a user-specified subset. VERIFY has basically the same error detection capabilities as CHECK, and similarly documents all errors for the user. VERIFY also has an automated correction capability similar to that of CHECK. The use of this mode is not recommended however, since most edit file features have already been determined to require interactive correction, and automated correction is therefore impossible or unreliable.

Program MERGE is the final step in the error correction process. It may be used interactively or in batch mode. MERGE assembles the corrected manuscript header file data, corrected edit file data, and the contents of the intermediate DFAD tape into a corrected output DFAD tape. Both the manuscript header file and the edit file are optional inputs. If neither is specified, there is no reason to perform the merge, since the output tape will be an exact copy of the intermediate tape used as input. The merge process is fully documented with summaries at both manuscript and tape level.

Internal Data Files

In addition to the eleven programs discussed above, several internal data files are produced. The first is the intermediate DFAD tape, which as defined previously is a copy of the original DFAD tape with automated corrections implemented.

The remaining files are disk resident. These include manuscript header files, review files, directory files, and edit files. Specific file names are created by the DFADEDIT programs based on user-supplied root names which are usually a function oı the volume number of the tape being corrected. Thus multiple tapes may be undergoing various phases of error correction concurrently.

The manuscript header file is a sequential file which contains an entry for each manuscript on the subject DFAD tape. A manuscript header file may be generated by either MANFILE or MANEDIT, and is used as input by MANEDIT and MERGE. Each manuscript entry contains all the manuscript header parameter values for that manuscript in both the original and modified forms.

The directory file is a random access file which contains an entry for each feature from the DFAD tape with one or more errors. The directory file entry contains feature header information which allows for unique identification of the feature, along with feature type and error descriptors. This file is generated by CHECK and is used as input to FETCH when generating the edit file. The directory file can be modified by the user with the DIRECTRY program.

The review file is a sequential file which is generated by CHECK. A review file may also be generated if VERIFY is run in automated correction mode. Review files contain features which have been corrected automatically, and are used only by REVIEW tc ·erify the correctness of automated corrections.

Finally, edit files are sequential files that normally are generated only by FETCH and EDIT. VERIFY will also generate edit files if used in automated mode. Edit files are used as input by EDIT, VERIFY and MERGE. They contain features which require interactive correction, or which have been corrected and are ready for verification or merge.

IV. Algorithm Development

A number of algorithms have been developed to detect and in some cases correct the various types of errors addressed by the DFADEDIT software. Some tests require considerable computation, while others involve simple range checks. The development and function of each of these algorithms is discussed in this chapter.

Manuscript and Feature Header Errors

The detection of erroneous parameters in manuscript and feature headers is based on comparison of the various header parameters with the valid ranges defined by the specification (Ref 2). The correction of header errors must be done interactively, since errors of this type are indicative of relatively severe problems. The software either reports out of range parameters as errors, or issues warnings to call potential problems to the attention of the user. The interactive edit capabilities provided in programs MANEDIT and EDIT allow the user to change any erroneous manuscript or feature header parameter. For more discussion, and for definitions of valid ranges for header parameters, refer to the DFADEDIT User's Guide in Appendix A.

Skewed Manuscripts

When the FAC 1 feature for each manuscript is processed, a skewed or non-rectangular manuscript test is performed. This test examines the four points which define the FAC 1 feature to determine if these latitude/longitude deltas define a rectangle whose edges are parallel to meridians of longitude or parallels of latitude. Although the specification allows a one-tenth arc-second skew, the software will report any skew as an error, since it's presence can lead to problems such as feature mis-registration across manuscript boundaries, feature points outside the manuscript, and gaps or overlaps in the coverage between adjacent manuscripts. Skewed manuscript boundaries can be corrected by the user by modifying the FAC 1 coordinates with the EDIT program.

Stacks, Spikes, and Open Areal Features

The detection and correction of stack points, spike points, and open areal features is straightforward and can be reliably performed automatically except for a very small number of occurrences.

Stack points are characterized by two consecutive feature points which are geographically identical. The correction for stack points is to delete one of the conflicting points. The only complication that can arise in this approach occurs when the point initially chosen for deletion is flagged as a manuscript exit point. An exit point flag is signified by a negative lattitude value, and its presence means that the feature exits the manuscript at this point. If the point to be deleted is flagged as an exit point, the other point of the identical pair is deleted.

Spike points occur when the first and third of three consecutive feature points are geographically equal. These errors are corrected automatically by deleting either the first and second, or the second and third of the three points involved. The second and third points are deleted unless the third point is an exit point. If the third point is an exit point, the first and second points are deleted.

When an areal feature is digitized during DFAD production, the digitizer attempts to choose an identifiable starting point from which he begins to trace the feature outline. When he nearly completes the feature, he stops just prior to the starting point. Software is then used to "close" the feature. This involves adding a point to the end of the feature that is geographically equal to the first point. This software closure was apparently never applied to a number of areal features in the data base. The correction of an open areal feature thus involves incrementing by one the number of feature points, and adding this new point to the end of the feature point list with latitude and longitude equal to that of the first point. This automated approach fails only if the feature is "open" and already has the 8191 maximum points allowable. If this is the case, the feature is flagged for interactive correction.

The correction of stack points, spike points, and open areal features is therefore always handled automatically except for the one possible complication with open areal features. It should be noted that an open areal feature with 8191 points was never encountered in all the data processed while implementing and testing the DFADEDIT software.

Zero Area and Clockwise Areal Features

Areal features must undergo two additional tests to verify that they bound a non-zero area and that they are digitized in a counterclockwise direction. The tests for these conditions require that the feature area be calculated.

The area of a polygon defined by n points, (x_1,y_1), (x_2,y_2)... (x_n,y_n), may be determined using the following formula (Ref 1:353):

$$A=\tfrac{1}{2}[x_1y_2+x_2y_3+\ldots+x_{n-1}y_n+x_ny_1-y_1x_2-y_2x_3-\ldots-y_{n-1}x_n-y_nx_1] \qquad (1)$$

This computation is performed for each areal feature, with the desired result expressed in square nautical miles. Since the input coordinates are in tenths of seconds of latitude/longitude, a conversion must be performed to arrive at area in square nautical miles. This conversion is accomplished using earth radius of curvature conversion factors for both latitude and longitude. These curvature conversion factors are calculated once per manuscript, based on the average of the minimum and maximum manuscript latitude defined by the manuscript boundary. The derivation of the curvature conversion factors and their application to feature area calculation is provided in Appendix C. Earth radius of curvature is further addressed by Pitman (Ref 4: App B).

Once the feature area is calculated using an implementation of Eq (1), the first step is to test for zero area. If the computed area is zero, an appropriate error message is issued, and the feature is set aside for interactive editing. A negative area indicates that the feature is digitized in a clockwise direction, in which case the feature is also flagged for interactive processing. The user can reverse the digitization of a clockwise feature with a single EDIT command. EDIT options for correcting a zero-area feature include modifying the feature, deleting it completely, or replacing it with a new feature created interactively.

Points Outside Manuscript

The test to determine if any feature points lie outside the manuscript involves a simple comparison of each point of the current feature to the manuscript boundary as defined by the FAC 1 feature. Any point which is positioned outside the FAC 1 limits is flagged as an error. If the FAC 1 feature is not perfectly rectangular, a skewed manuscript exists. In this case, the point outside manuscript test is performed using the smallest area rectangle defined by the FAC 1 feature. This insures that no points outside the manuscript will be overlooked if a skewed manuscript is reduced to the smaller rectangular size during error correction. Features with points outside the manuscript must be corrected interactively.

Intersections

Intersections are one of the most common types of errors, and they are also one of the most potentially damaging. The presence of intersections was the primary reason for the initial development of the error correction software. In the discussion that follows, the general intersection detection algorithm will first be developed. Then the discussion will turn to the classification of intersections into seven basic types, and the automated correction of each type.

Intersection Detection. An intersection error occurs whenever two non-consecutive line segments of a linear or areal feature intersect, or whenever two consecutive line segments intersect at some point other than their common endpoint. The concept demonstration version of this software could detect intersections automatically, but all corrections were performed interactively. The enhanced version of the DFADEDIT software, implemented at DMA during this effort, has the capability to detect all intersections and automatically corrects a high percentage of such errors.

The test to determine if a feature contains intersecting lines is applied to every line segment pair of a feature. That is, every feature line segment must be tested with every other feature line segment to completely test for intersections. The basic algorithm for determining whether a line segment pair intersects requires that the equations of the two lines be derived in general form. The general form of a line is expressed as:

$$Ax + By + C = 0 \tag{2}$$

This general form can be derived from the two point form,

$$y-y_1 = \frac{y_2-y_1}{x_2-x_1} (x-x_1) \tag{3}$$

as follows. Equation 3 can be rewritten as

$$\frac{y_2-y_1}{x_2-x_1} x - y + y_1 - \frac{y_2-y_1}{x_2-x_1} x_1 = 0 \tag{4}$$

From equation 4, the following values of A, B, and C for the general form can be derived:

$$A = \frac{y_2-y_1}{x_2-x_1} \tag{5}$$

$$B = -1 \tag{6}$$

$$C = y_1 - \frac{y_2-y_1}{x_2-x_1} x_1 = y_1 - Ax_1 \tag{7}$$

These values of A, B, and C can be computed as long as $x_1 \neq x_2$. If $x_1 = x_2$, the line is vertical, and division by zero results from the attempt to calculate A and C. The algorithm handles this case by making the following assignments when $x_1 = x_2$:

$$A = 1 \tag{8}$$

$$B = 0 \tag{9}$$

$$C = -x_1 \tag{10}$$

The equations of the two line pairs may be expressed in general form:

$$A_1x + B_1y + C_1 = 0 \tag{11}$$

$$A_2x + B_2y + C_2 = 0 \tag{12}$$

According to Gilloi (Ref 3:157), if the determinant of these lines is equal to zero,

$$\begin{vmatrix} A_1 & B_1 \\ A_2 & B_2 \end{vmatrix} = 0 \tag{13}$$

the lines are parallel. If the lines are parallel, a further test must be performed to see if they are coincident, since coincident lines are a type of intersection. According to Giloi, if

$$\frac{A_1}{A_2} = \frac{B_1}{B_2} = \frac{C_1}{C_2} \tag{14}$$

the lines are coincident and an intersection error has been detected.

If the lines are not parallel, their intersection point (x_i, y_i) is calculated:

$$x_i = \frac{B_1C_2-B_2C_1}{A_1B_2-A_2B_1} \tag{15}$$

$$y_i = \frac{C_1A_2-C_2A_1}{A_1B_2-A_2B_1} \tag{16}$$

If (x_i,y_i) lies within the defined segments for both lines of the line pair under consideration, the line segments intersect. Assume that the first line has endpoints P1 = (x_1,y_1) and P2 = (x_2,y_2), and that the second line has endpoints P3 = (x_3,y_3) and P4 = (x_4,y_4). Then the two segments P1,P2 and P3,P4 intersect if, by Giloi (Ref 3:157),

$$\max[\min(x_1,x_2),\min(x_3,x_4)] \le x_i \le \min[\max(x_1,x_2),\max(x_3,x_4)] \tag{17}$$

and

$$\max[\min(y_1,y_2),\min(y_3,y_4)] \le y_i \le \min[\max(y_1,y_2),\max(y_3,y_4)] \tag{18}$$

Max and min here refer to the maximum and minimum values, respectively, of the items within the associated parentheses.

The first implementation of the intersection test was a brute force, exhaustive algorithm that performed every line comparison completely through the calculation of the intersection point and segment test. A five to ten factor increase in algorithm speed was realized by implementing a preliminary box test for each line pair prior to computing the full range of detailed intersection tests. The box test is illustrated in Figure 5. The dashed lines represent the imaginary "boxes" constructed by the intersection detection algorithm to perform this test. A "box" is constructed around each of the two line segments being compared using the coordinates of the endpoints of each segment. If the

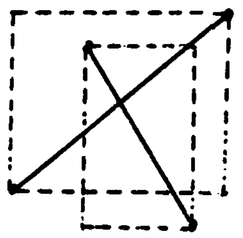


Fig 5-a. Box Text - Boxes Intersect,
Line Segments Intersect

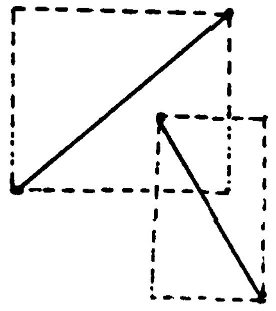


Fig 5-b. Box Test - Boxes Intersect,
Line Segments Do Not Intersect

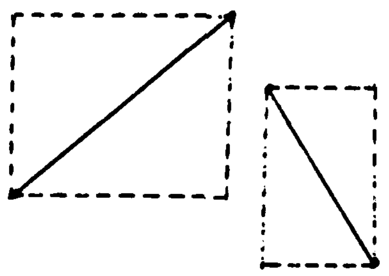


Fig 5-c. Box Test - Boxes Do Not Intersect,
Line Segments Do Not Intersect

boxes intersect, (Figures 5-a,b), the detailed intersection test must be performed since the line segments may or may not intersect. If however the boxes do not intersect (Figure 5-c), the line segments cannot intersect, and no further tests on the current line segment pair need be done. This leads to a substantial savings in the CPU time requirements for intersection detection.

<u>Automated Correction</u>. The first step in the automated correction of an intersection involves classifying the intersection as one of seven basic types. Once the type of intersection has been determined, the software will attempt to resolve the intersection either by deleting a selected feature point, or by rearranging the numbering of two or more feature points.

Seven basic categories of intersections are addressed by automated correction. These include coincident points, closure intersections, coincident lines, compressed feature intersections, irregular spikes, point-line intersections and general intersections. Examples are provided as the details of automated correction for each type are discussed below.

Coincident points are the first intersection type tested for. Each feature point is tested against all other feature points. A coincident point intersection is illustrated in Figure 6-a, with the feature points numbered as shown. When such an error is detected, the distances d_1 and d_2 shown in Figure 6-b are computed. Since the automated correction for the resolution of coincident points requires the deletion of one of the points involved, a user control parameter called DELETDST, specified in tenths of seconds, is compared to the distances d_1 and d_2.

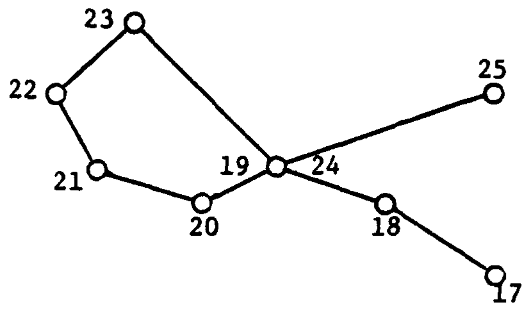


Fig 6-a. Coincident Point Intersection

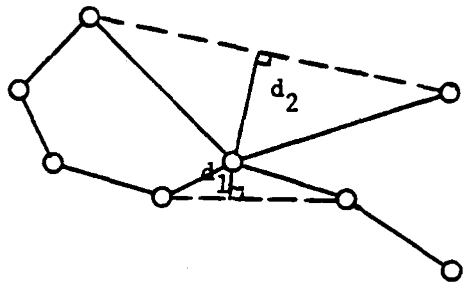


Fig 6-b. Calculation of Deletion Distances

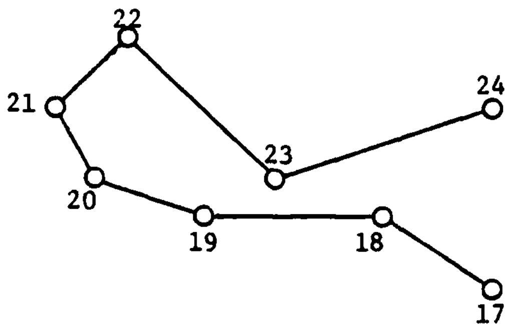


Fig 6-c. Coincident Point Intersection Resolved

If neither d_1 or d_2 is less than DELETDST, the feature is flagged for interactive correction. If either is less than DELETDST, the point associated with the smallest distance is deleted. For this example, assume that $d_1<d_2<$DELETDST. Point 19 is thus deleted, and the result is shown in Figure 6-c. This approach controls and minimizes feature distortion due to automated correction.

Closure intersections involve the closure of an areal feature, and thus are usually special cases of coincident line or general type intersections. The general case closure intersection is illustrated in Figure 7-a. In this case, the second-last line of a feature intersects the first line of the feature. When this occurs as depicted in the example, the distance from point 6 to point 7 is calculated. If this distance is less than DELETDST, point 6 is deleted. If this distance is greater than DELETDST, interactive editing is required. The automatically corrected result is shown in Figure 7-b. This type of correction is not normally applied to a general type intersection as discussed below, however, it is the most appropriate correction where feature closure is involved. The other type of closure error, which is much less common, is illustrated in Figure 7-c. This case is indicative of a more severe digitization error and thus no automated correction is ever attempted.

A compressed feature intersection is shown in Figure 8. This error is encountered along a manuscript boundary where a portion of a feature has been compressed to two colinear segments. This risk associated with automated correction of this type of intersection is considered high, and thus this error type is always flagged for interactive

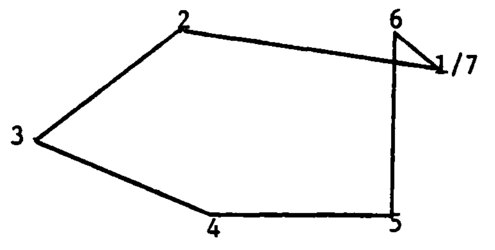


Fig 7-a. General Closure Intersection

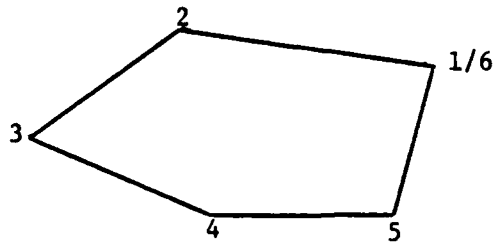


Fig 7-b. General Closure Intersection Automatically Resolved

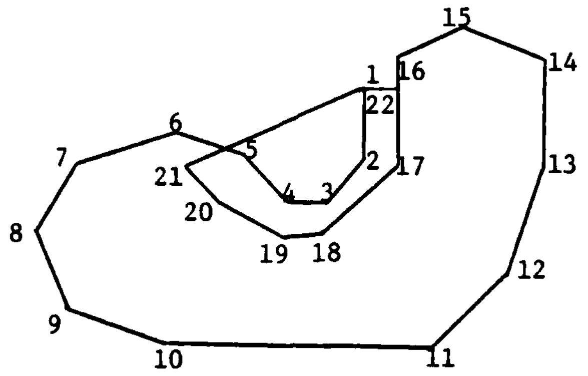


Fig 7-c. Severe Closure Intersection

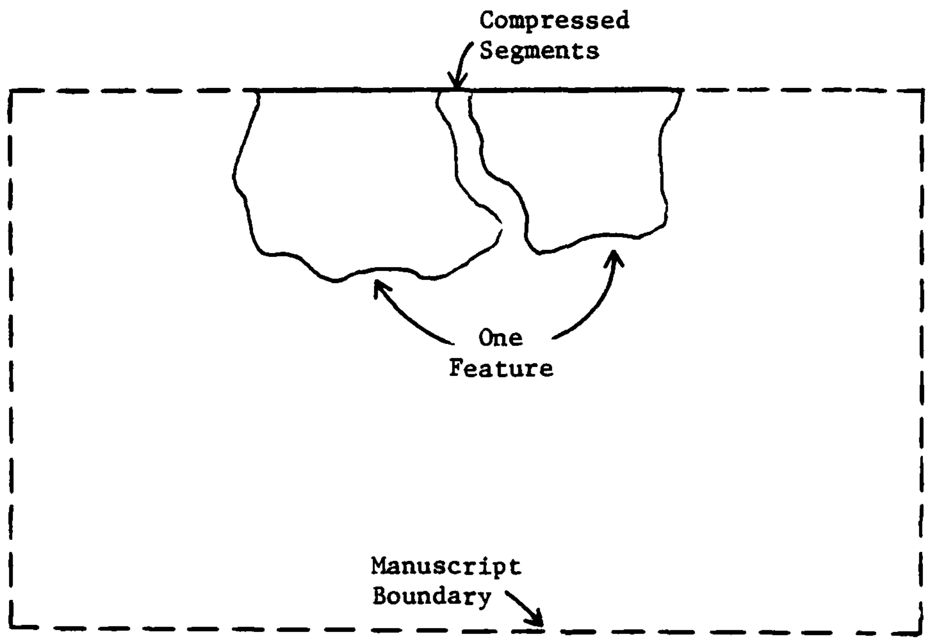


Fig 8. Compressed Feature Intersection

correction. This is the only type of intersection which is never corrected automatically.

A coincident-line intersection is illustrated in Figure 9-a. The first step of automated correction for this type of intersection involves computing the four distances d_1, d_2, d_3, and d_4 shown in Figure 9-b. If the shortest of these distances is less than DELETDST, the associated point is deleted. This sometimes leaves the result shown in Figure 9-c, which is a point-line intersection. This type of intersection is resolved as discussed below and illustrated in Figure 11. If d_1, d_2, d_3, and d_4 are all greater than DELETDST, the feature must be interactively corrected.

An irregular spike intersection is illustrated in Figure 10-a. The first step in resolving this error involves computing distance d. If d is less than DELETDST, point 12 is deleted automatically. The result is shown in Figure 10-c. If d is greater than DELETDST, once again the feature must be corrected interactively.

The next type of intersection is the point-line intersection, illustrated in Figure 11-a. If distance d (Figure 11-b) is less than DELETDST, point 47 in the example is deleted. The result is shown in Figure 11-c. Interactive correction is required if d > DELETDST.

The final and most common type of intersection is the general case shown in Figure 12-a. If an intersection is detected and it does not match any of the other types discussed above, it is assumed to be a general intersection. Automated correction of general intersections requires computation of distance d in Figure 12-b. This distance is compared to the user control parameter UNWINDST. If d < UNWINDST, the

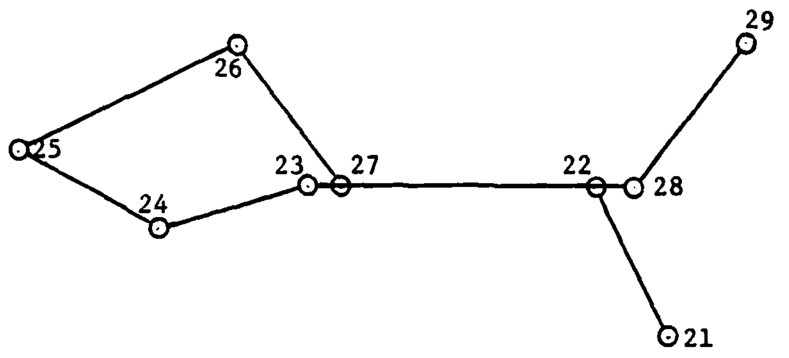


Fig 9-a. Coincident Line Intersection

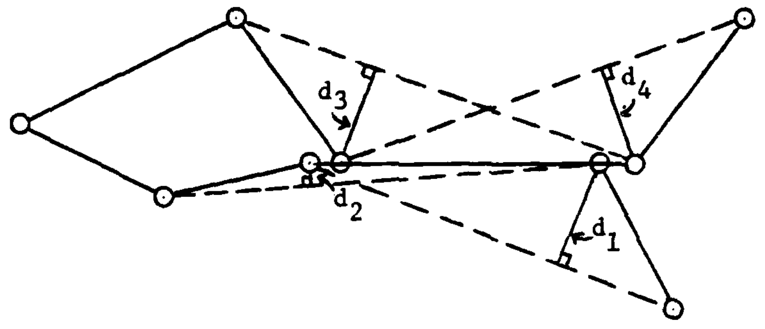


Fig 9-b. Deletion Distance Calculation

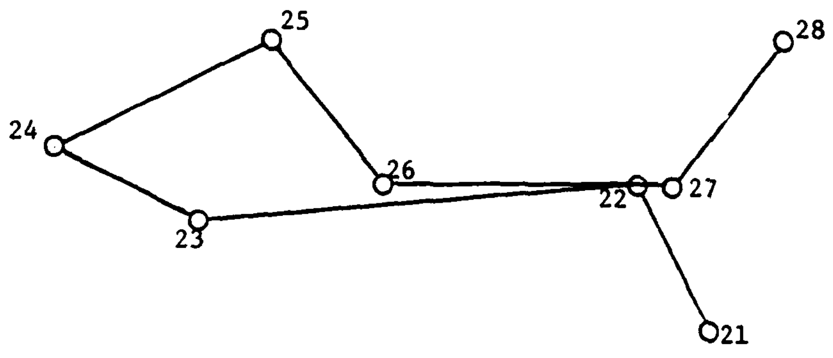


Fig 9-c. Coincident Lines Automatically Resolved, Resulting in a Point-Line Intersection

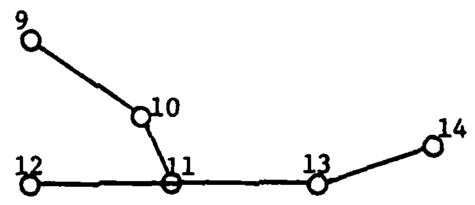


Fig 10-a. Irregular Spike Intersection

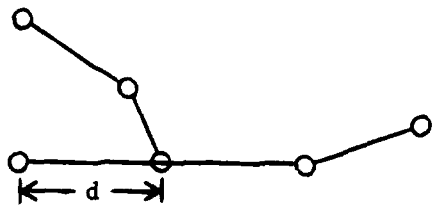


Fig 10-b. Deletion Distance Calculation

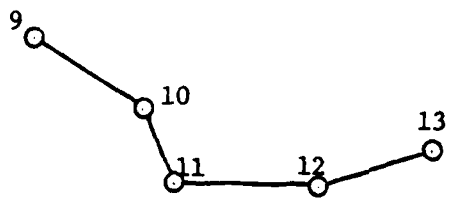


Fig 10-c. Irregular Spike Automatically Resolved

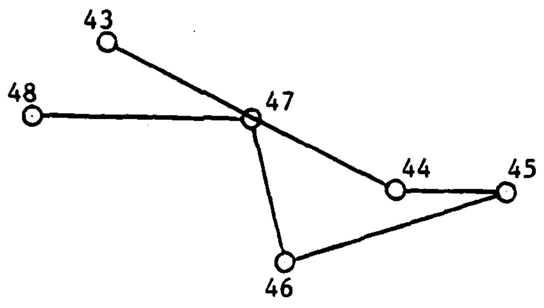


Fig 11-a. Point-Line Intersection

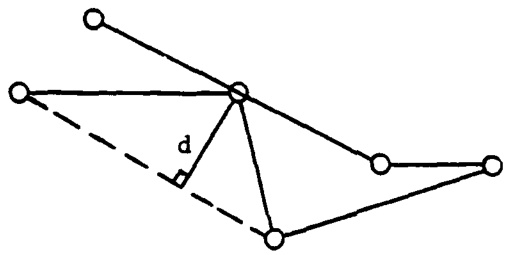


Fig 11-b. Deletion Distance Calculation

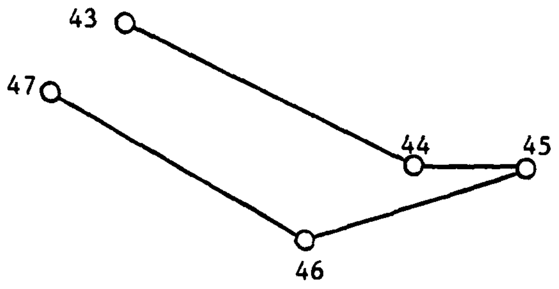


Fig 11-c. Point-Line Intersection Automatically Resolved

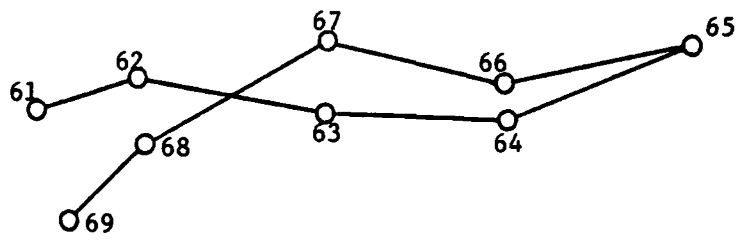


Fig 12-a. General Intersection

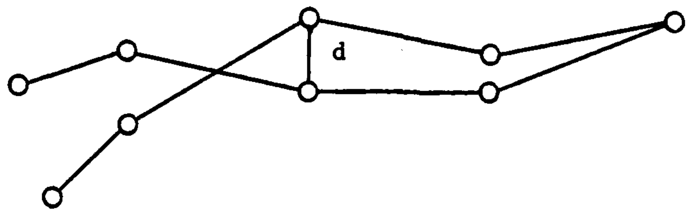


Fig 12-b. "Unwind" Distance Calculation

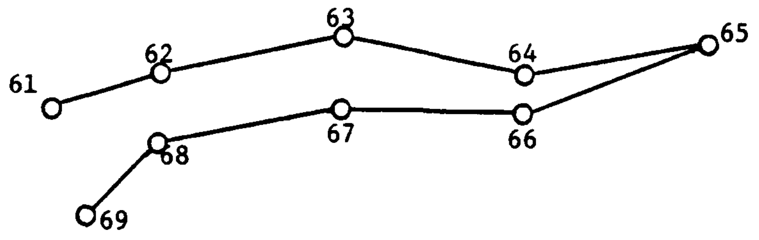


Fig 12-c. General Intersection Resolved

intersection is resolved automatically by "unwinding", or renumbering, the affected points in the vicinity of the intersection. The result is shown in Figure 12-c. Note that this correction involves no point deletion, and thus fully preserves feature detail. If d is greater than UNWINDST, interactive correction is once again required.

In addition to the DELETDST and UNWINDST control parameters, the user employs several other parameters which control the automated correction algorithms. AUTOMAX controls the maximum number of automated intersection and exit point errors corrected for a single feature. LFPNTVAR is specified as a percentage, and controls the variation in the number of points of a linear feature due to automated intersection correction. AFPNTVAR performs the same control function for areal features. AFAREVAR controls the percent variation allowed in areal feature area due to the automated correction of intersections. All of these parameters are used by the CHECK program to control automated error correction. If any of these parameters is exceeded, the subject feature is restored to its original configuration and flagged for interactive correction.

The full range of interactive EDIT commands is available to assist the user in the correction of any intersection errors which cannot be handled automatically. The user may interactively unwind general intersections, or he may delete, change, insert, or move points. The capability to fully analyze and correct all intersection errors is provided.

Exit Point Errors

Manuscript exit points are those points in a feature where the path or boundary of a linear or areal feature, respectively, leaves the manuscript. These points are signified in the DFAD by a negative latitude. An exit point logically then must lie on the manuscript boundary defined by the FAC 1 feature. Errors occur in conjunction with manuscript exit points in one of two ways.

The most common exit point error involves a point that geographically is a valid exit point, but is not so flagged. The straightforward automated correction in this case involves simply reversing the sign of the latitude of the subject point.

A much less frequent but more severe exit point error involves a point which geographically is not an exit point, but is flagged with a negative latitude. This error occasionally occurs on skewed manuscripts, but is considered too serious for automated correction.

It should be noted that any automated corrections already performed on a feature are discarded if an error requiring interactive correction is discovered. The user must perform all corrections on such features interactively.

Warnings

A number of special tests are performed for conditions which are not always indicative of errors, but which are suspicious enough to require interactive inspection by the user. When such conditions are encountered, an appropriate warning is issued, and the feature receives an error severity level representative of a potential error condition. These warnings are the subject of this section.

Average Length Spikes. Warnings are issued for features with points which display relatively large geographic displacements from the surrounding feature points. This error is illustrated in Figure 13, and is called an average length spike because of the method used to detect the error. Although these spikes occur infrequently, they can severely distort a feature. They may or may not involve intersections. The test for this type of error involves first computing the average segment length by considering all segments which make up the feature. Each point is then tested by comparing the length of the segment on each side of the subject point to a user-defined multiple of the average length. If the two surrounding segments (I-J and J-K) are both longer than the multiple of the average length, a test is performed to determine if the two points (I and K) adjacent to the test point (J) are within a user defined distance of each other. If so, a warning is issued to alert the user to the possibility of an average length spike at point J.

Wrap-Around Features. Another type of warning is issued for "wrap-around" features. As shown in Figure 14, this test is applied to find features which erroneously cover large parts of a manuscript and improperly cover up features with lower FAC numbers. The test for this condition involves checking each areal feature with FAC greater than 2 to determine if it touches 3 or more corners of the manuscript. In the example, FAC 160 is improperly portrayed, and thus overlays all features with lower FAC numbers. If such a feature is located, the user must decide whether the portrayal is valid or whether the problem results from a digitization or FAC hierarchy error.

Potential Exit Point Errors. The test for potential exit point errors is performed only on linear and areal features which reside on a skewed manuscript. A skewed manuscript is depicted in Figure 15-a. The

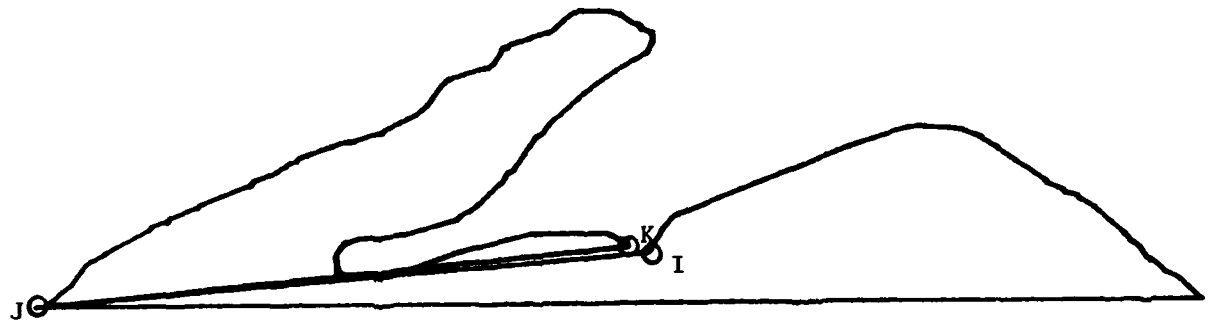


Fig 13. Average Length Spike

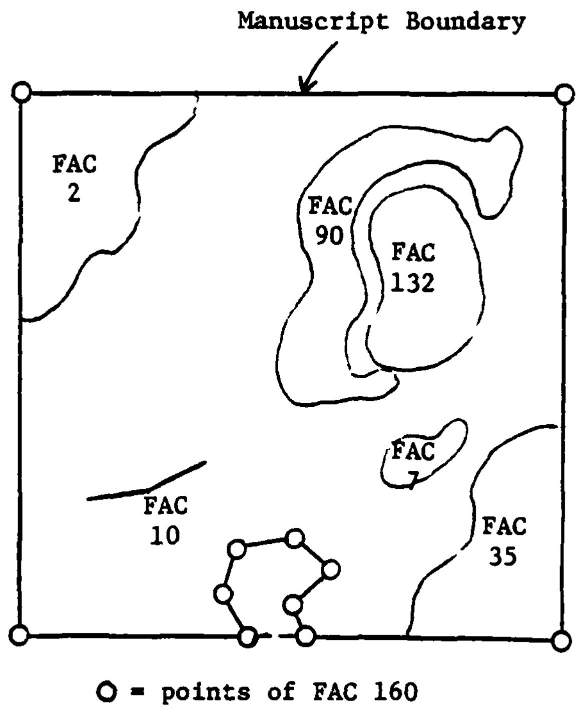


Fig 14. Wrap-Around Features

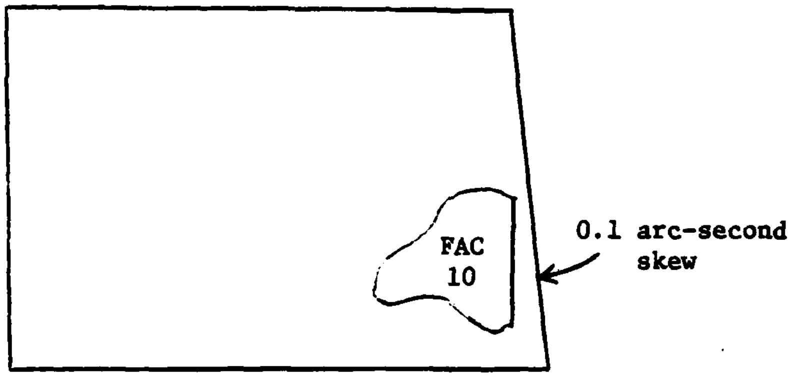


Fig 15-a. Potential Exit Point Errors-Skewed Manuscript

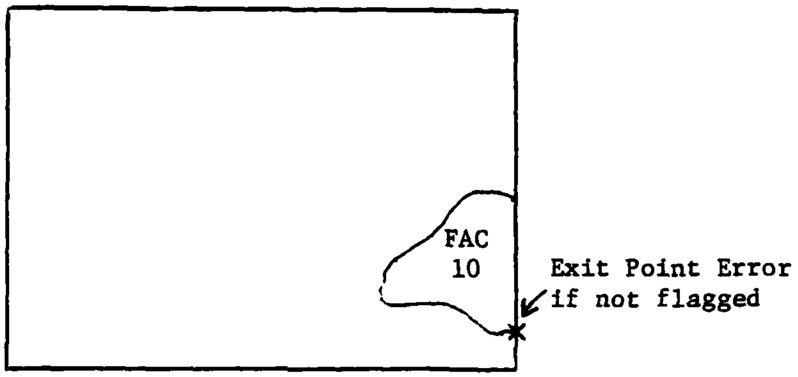


Fig 15-b. Exit Point Error Eliminated When Skew Corrected

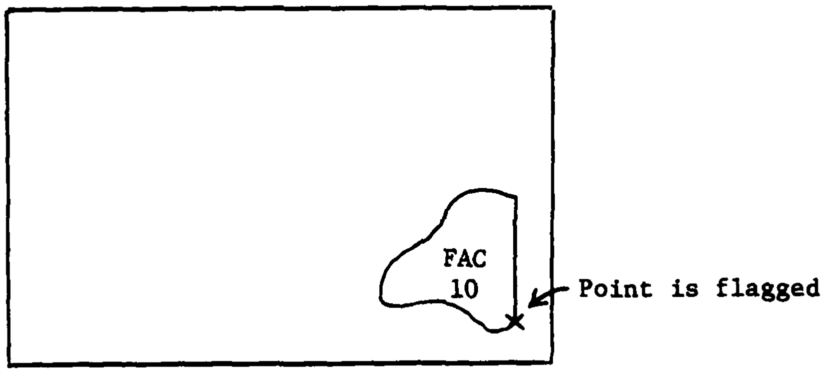


Fig 15-c. Exit Point Error Remains When Skew is Corrected

test assumes that the correction of a skewed manuscript will invôlve modification of the skewed manuscript boundary to one of the representations shown in Figures 15-b,c. Assume that FAC 10 in Figure 15-b has no points flagged as exit points. If the boundary skew is corrected as shown in Figure 15-b, FAC 10 should have a point flagged as an exit point in the location shown. However, if the exit point is properly flagged but the skew is corrected as shown in Figure 15-c, an error still results. The potential exit point error test is thus necessary to insure one-pass correction. It should be noted however, that changing the manuscript boundary to some other configuration which the software cannot predict could introduce exit point errors which would not become evident until the corrected output tape from MERGE was re-checked.

Closed Linear Features. A test is performed to check linear features for closure. If the last point of a linear feature is geographically identical to the first, the feature is said to be "closed". While this is not necessarily indicative of an error, it should be examined by the user to be certain that the feature is properly portrayed, and is not a miscoded areal feature.

Point and Linear Feature Point Warnings. Finally, a warning may be issued for a point feature which contains more than a user-specified (PFPNTWRN) number of points. A similar warning is issued for linear features, and is controlled by a separate user parameter (LFPNTWRN). These may be false indications of errors, but occassionally a point or linear feature with a large number of points is indicative of a problem.

V. VAX Implementation and Test

This chapter addresses special topics with respect to the DFADEDIT software implementation on the VAX. Test methodology and results are discussed, and an example which demonstrates the effectiveness of automated intersection correction is presented.

VAX Implementation

The VAX implementation of the software is consistent with the architecture presented in Figure 4. All eleven programs have been implemented and fully tested. All the programs run interactively, and CHECK, VERIFY, and MERGE can also be run in batch mode due to their relatively long execution times. Each program generates a listing file which documents user control parameters, interactive modifications, diagnostic information, or corrective action taken by the program, as applicable. Details concerning the actual operation of individual programs are contained in the DFADEDIT User's Guide, provided in Appendix A. System level flow charts are provided in Appendix D.

Some discussion of the execution time involved in the CHECK, EDIT, and VERIFY programs when performing intersection detection is in order. Intersection detection may require from less than 1 second for small features up to 40 minutes for features with 8000 or more points. In the worst case, the time required could be nearly doubled if, when performing automated correction, an intersection requiring interactive correction is detected near the end of automated intersection processing for the feature. If this occurs, all automated corrections are cleared, the feature delta set is restored to its original form, and processing of the feature starts over in intersection "detection-only" mode. In this way, all feature errors are documented to allow for interactive correction. However, the processing time required is almost twice the normal intersection processing time. Although this is not a common occurrence, it is bound to happen occassionally. Fortunately, most manuscripts have only a few, if any, features with more than 1000 points, and features of 5000 or more points are relatively rare.

Software Test

The DFADEDIT software as implemented on the VAX has been thoroughly tested for its ability to detect and correct errors of the types addressed in the System Requirements Definition. A test tape was developed by modifying an actual DFAD tape using the DFADEDIT programs. The test tape contains eight manuscripts, seven of which contain various errors. One manuscript contains no errors. The tape contains one or more occurrences of most types of errors addressed by the software. In addition, special cases were inserted to allow some test features to pass or fail automated intersection correction depending upon the values of the user supplied control parameters DELETDST, UNWINDST, LFPNTVAR, AFPNTVAR, and AFAREVAR. The test tape was processed predictably and correctly in all cases, and all capabilities of all the DFADEDIT programs were utilized and successfully tested.

Automated Correction Example

This section demonstrates the capability provided by automated correction. Figure 16-a depicts a complex areal feature with 7718 points which portrays a forest. When this feature was digitized, apparently an attempt was made to trace around some very fine detail (possibly streams) in the right half of the feature. In so doing, 158 separate intersections were introduced. At the time this feature was originally encountered, automated correction was not yet developed, and CHECK required about two hours to simply detect the intersections. Approximately six additional hours were spent using the EDIT program to interactively eliminate these errors. Even with this excessive time expenditure, it was necessary to delete a large number of points, and the fidelity of the feature portrayal was noticeably reduced. Figure 16-b shows the feature after interactive correction.

With the development of automated correction, this feature can be corrected in about the same amount of time required by the current software to detect the intersections. The automatically corrected feature as shown in Figure 16-c was processed by the DFADEDIT CHECK program in approximately 25 minutes, and all intersections were resolved. Not only is automated correction much faster in this case, but the feature detail is very well preserved since only 17 points were deleted. Although this may be an extreme example of the effectiveness of automated correction, numerous other erroneous features can be corrected just as effectively, with proportionate savings in time and fidelity.

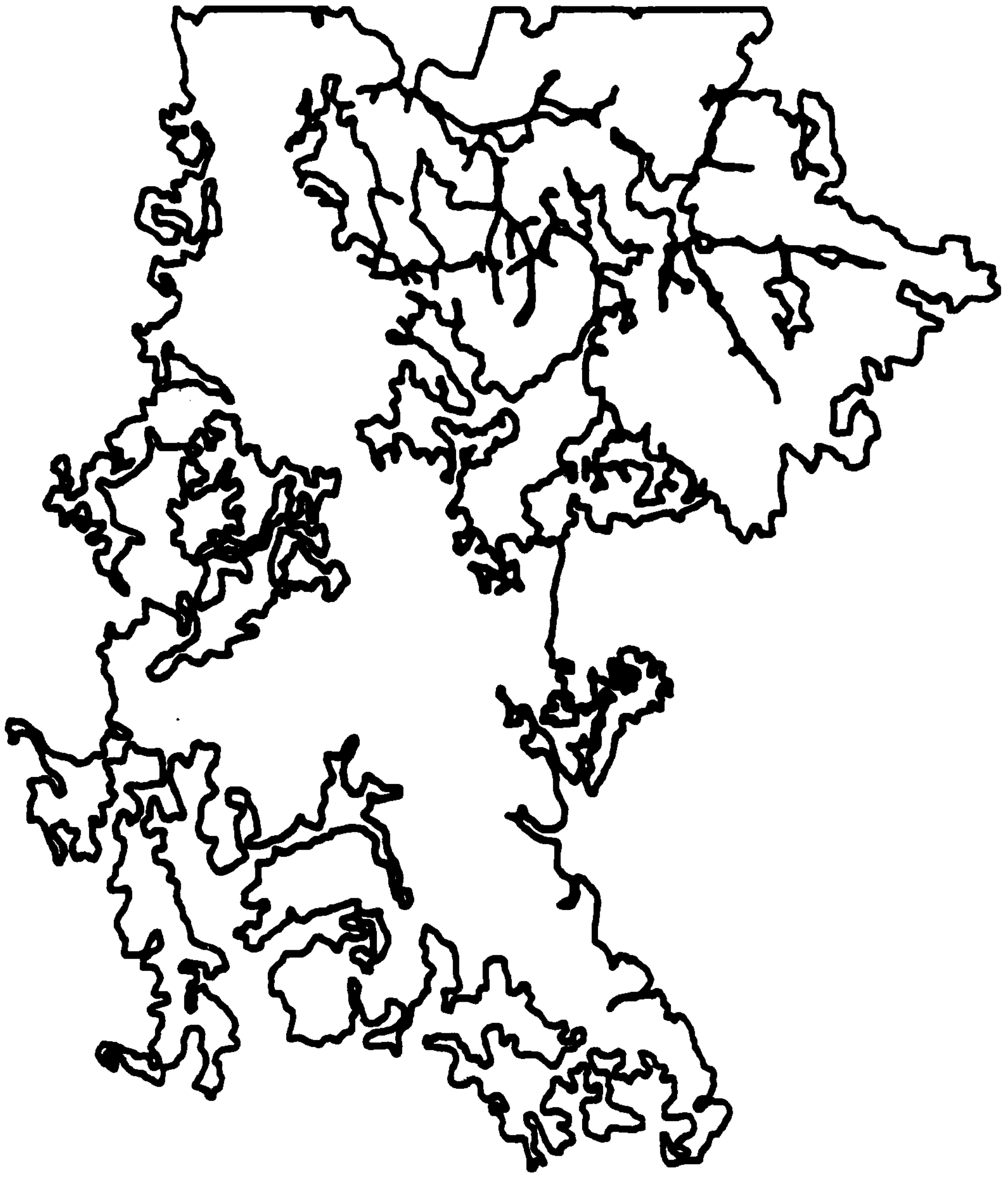


Fig 16-a. Original Feature With 7718 Points

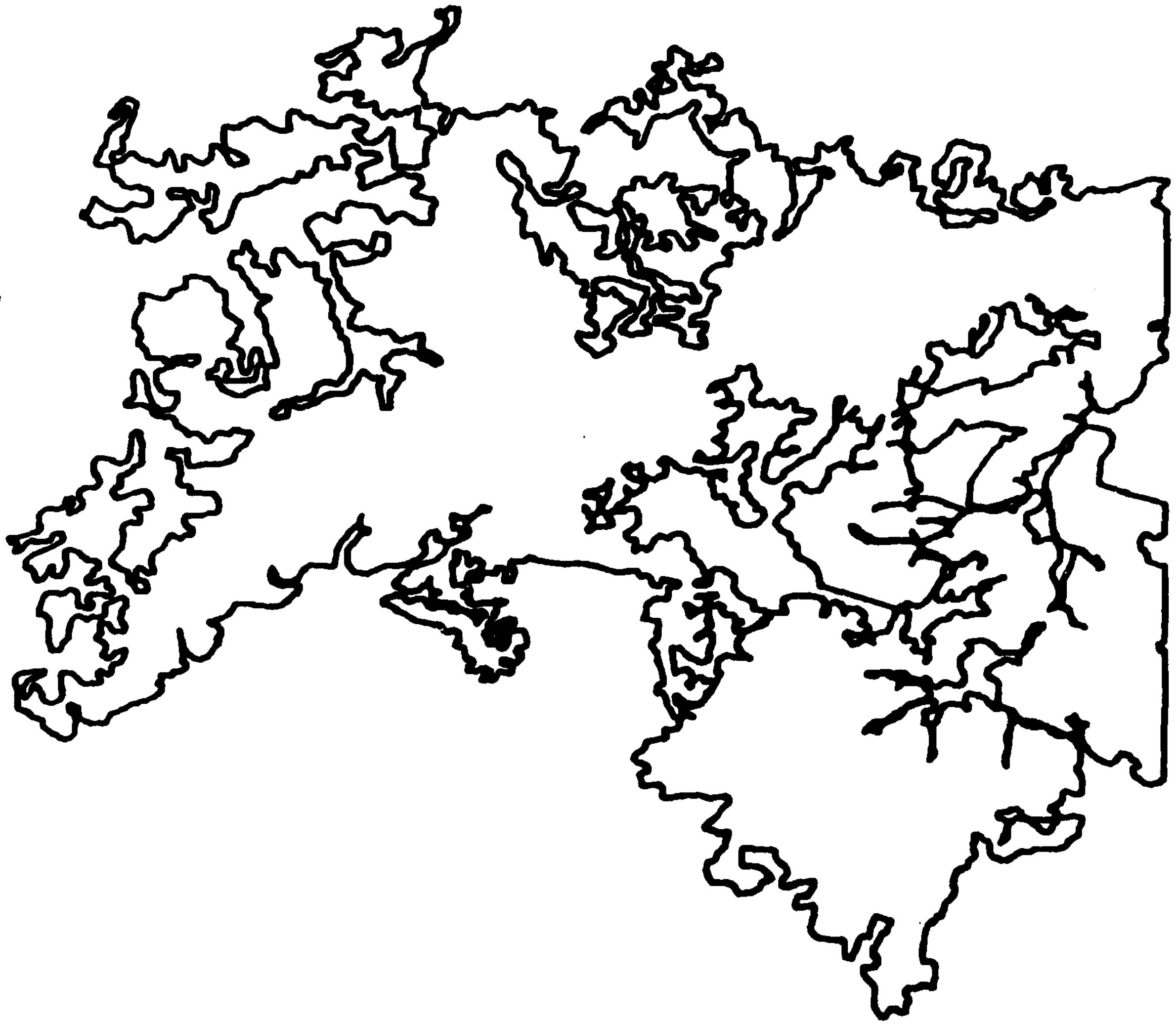


Fig 16-b. Interactively Corrected Feature With 7081 Points

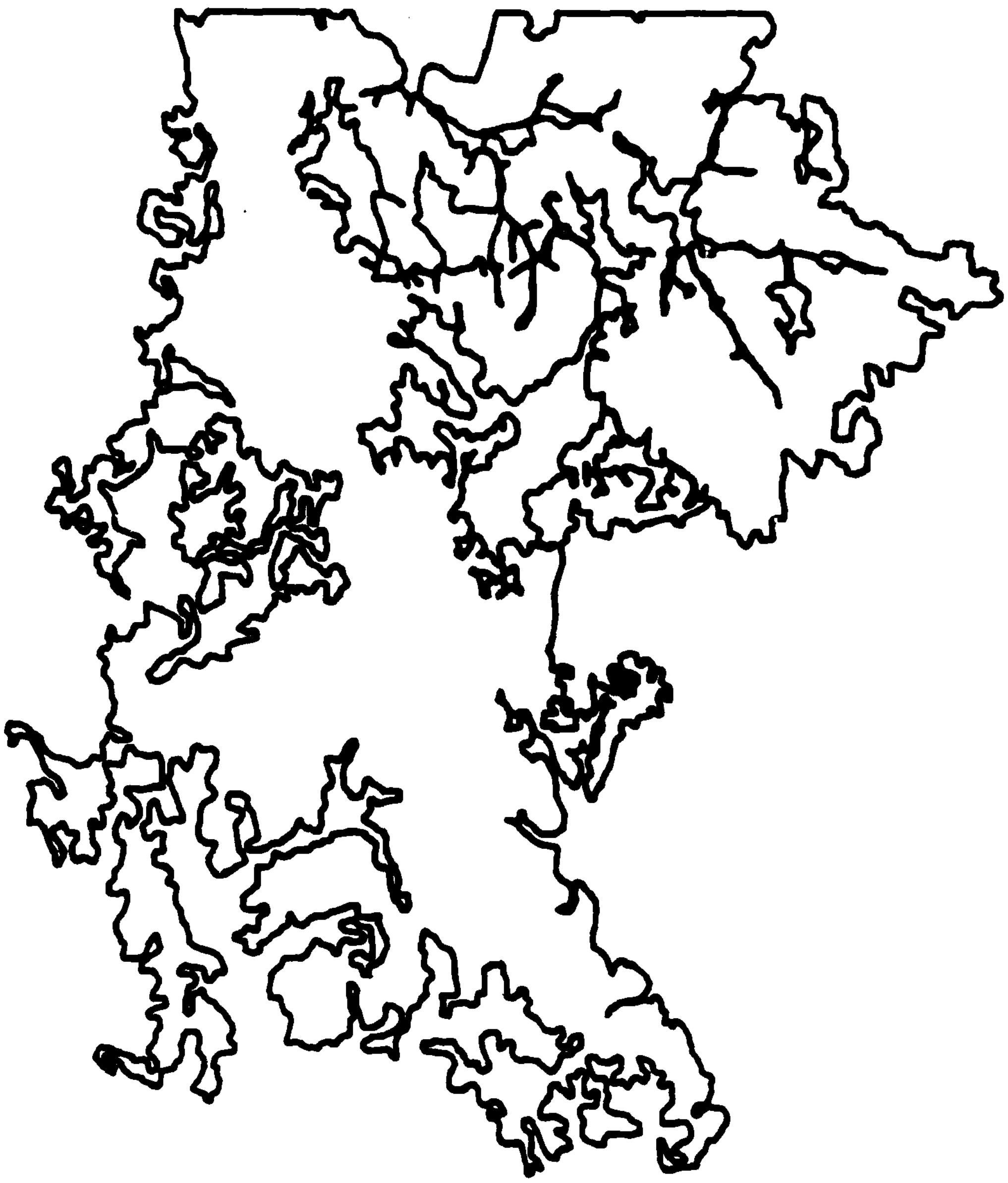


Fig 16-c. Automatically Corrected Feature With 7701 Points

VI. Conclusions and Recommendations

Two objectives were defined for this project. The primary objective was to design and implement a software package that could detect and provide the capability to correct a number of errors commonly found in the DMA DFAD digital cartographic files. The secondary objective was to implement automated correction procedures for certain types of errors, if feasible. This chapter briefly summarizes the results of the development of the DFADEDIT software, and provides some recommendations for future consideration or implementation.

Conclusions

Both of the objectives mentioned above were met. The DFADEDIT software detects all occurrences of those errors discussed in Chapter II. Methods are provided whereby each identified error type can be corrected, either automatically or interactively, and an error-free DFAD tape can be reliably generated in one pass through the DFADEDIT software.

Automated correction capability has been proven feasible for some types of errors. Stack points, spike points, and open areal features are relatively common errors, and the automated correction of such errors is considered low-risk. Virtually 100% automated correction has been attained for these three types of errors.

Intersections and exit point errors are two other relatively frequent error types. Automated correction of these errors is relatively high risk since some occurrences cannot be automatically corrected at all, and some intersection errors may pass automated correction with unacceptable results. The risk is eliminated by providing the user with sufficient controls over the automated correction process, and the ability to intercept and override unacceptable automated corrections.

Several deficiencies still exist in the DFADEDIT software. These include the excessive CPU time required to check complex features for intersections, the registration or matching of real-world features which cross manuscript boundaries, and detection of gaps or overlaps between adjacent manuscripts.

Recommendations

From earlier discussion in Chapter IV, it is clear that significant improvement has been made in the efficiency of the intersection detection algorithm. However, this algorithm is still the major contributor to the CPU time requirements of the DFADEDIT software. The payoff from improvements to this algorithm would be realized by increased efficiency in the CHECK, EDIT, and VERIFY programs, since all three make extensive use of intersection detection. Investigations on the optimization of the intersection algorithm should consider methods of reducing the number of line segment pairs tested by sorting the feature line segments by latitude or longitude. Further investigation should also consider the possibility of using different algorithms for various feature sizes or complexities, based on the number of feature coordinates. There is no guarantee that a more efficient, reliable algorithm can be developed, but the potential payoff makes some investigation worthwhile.

A second deficiency arises from the fact that the current DFADEDIT software deals with the DFAD on a single feature basis. That is, the software is limited to testing for errors within a feature, and testing the feature with respect to the containing manuscript. Little consideration has been given in the current software to the interaction between manuscripts. The continuity of real-world features across manuscript boundaries is of concern to DMA. The detection of mismatches, known as panelling errors, is a complex problem since features from separate manuscripts must be available simultaneously for comparison. The problem is compounded by the fact that adjacent manuscripts are not always contained on the same DFAD source tape. The current process used by DMA to correct these errors involves moving partially corrected data to another computer to detect the errors, then moving the data back to the VAX for interactive correction of the errors. This adds a great deal of time and overhead to the error correction process.

Implementation of techniques to address these panelling errors should be a high priority task. The integration of an additional program to detect these errors within the existing DFADEDIT software should be a straightforward process. A second program may be required to merge entries for features with panelling errors into the directory file. Once this directory file merge is completed, the features could be included in the edit file and processed like any other feature which requires interactive correction. One additional concern in implementing a panelling error capability is the user graphics terminal. A higher resolution terminal or hard copy plotting capability may be required.

A final topic for future consideration is the capability to automatically detect gaps or overlaps between adjacent manuscripts. Although the software now provides manuscript origin and corner point information, the user must manually verify that adjacent manuscripts join properly. The software solution to this problem is not simple, since adjacent manuscripts may be of varying size or may be slightly offset in latitude or longitude. However, this manual analysis is one of the few tedious tasks that remains in the DFADEDIT software, and an automated solution should be considered.

BIBLIOGRAPHY

1. Chemical Rubber Company. Standard Mathematical Tables. Cleveland, Ohio: The Chemical Rubber Company, 1971.

2. Defense Mapping Agency. Product Specifications for Digital Landmass System (DLMS) Data Base. St. Louis, Missouri: Defense Mapping Agency Aerospace Center, 1977.

3. Giloi, Wolfgang K. Interactive Computer Graphics. Englewood Cliffs, New Jersey: Prentice Hall Inc., 1978.

4. Pitman, George R. Inertial Guidance. New York: John Wiley and Sons Inc., 1962.

APPENDIX A

DFADEDIT User's Guide

DIGITAL FEATURE ANALYSIS DATA EDIT

(DFADEDIT) SOFTWARE

USER'S GUIDE

Written by:
Michael R. Nicol
USAF Aeronautical Systems Division
ASD/ENETV
Wright-Patterson AFB, OH 45433

Revision: C 07-SEP-82

TABLE OF CONTENTS

List of Figures

List of Tables

I. Introduction

1.1 Document Purpose

This document has been developed to assist the user in applying software developed by the U.S. Air Force Aeronautical Systems Division (ASD) to the correction of certain anomalies or errors in Defense Mapping Agency (DMA) Digital Feature Analysis Data (DFAD). The software package is collectively entitled DFADEDIT. This document addresses the use of DFADEDIT as implemented on DMA's VAX-11/780 computer system. It is assumed that the user has a working knowledge of the VAX/VMS operating system and its command language. For information concerning any operating system function or command, the user should refer to the appropriate VAX/VMS manual.

1.2 Specification Revision Levels

This software is compatible with DMA DFAD cultural data as defined in "Product Specifications for Digital Landmass System (DLMS) Data Base", up to and including Change 2, issued 25-AUG-80.

1.3 Software Revision Levels

This document is compatible with the eleven programs which comprise the DFADEDIT software package at the following revision levels:

Program	Version	Date
*******	*******	****
DFADSTAT	3.0	20-JUL-82
DFADPLOT	3.0	10-JUL-82
CHECK	3.0	20-JUL-82
MANFILE	3.0	10-JUL-82
MANEDIT	3.0	10-JUL-82
REVIEW	3.0	20-JUL-82
DIRECTRY	3.0	20-JUL-82
FETCH	3.0	20-JUL-82
EDIT	3.0	20-JUL-82
VERIFY	3.0	20-JUL-82
MERGE	3.0	20-JUL-82

II. Software Overview

2.1 General Description

The DFADEDIT software package is made up of eleven programs which are used in various combinations to detect and correct errors on selected manuscripts of a DMA DFAD input tape. Some types of errors may be corrected automatically if so desired by the user. The nature of other types of errors necessitates interactive correction by the user, regardless of whether automated correction is specified. Figure 1 depicts the process flow for automated error correction.

Each of the programs and the intermediate tapes and files generated and used during the error correction process are discussed briefly below. More detail on the various programs is contained in Chapters 3 through 13.

2.2 Errors Addressed

The DFADEDIT software was originally developed to address only a few specific errors. Once the concept of error detection/correction was demonstrated, it became clear that the approach was flexible enough to support correction

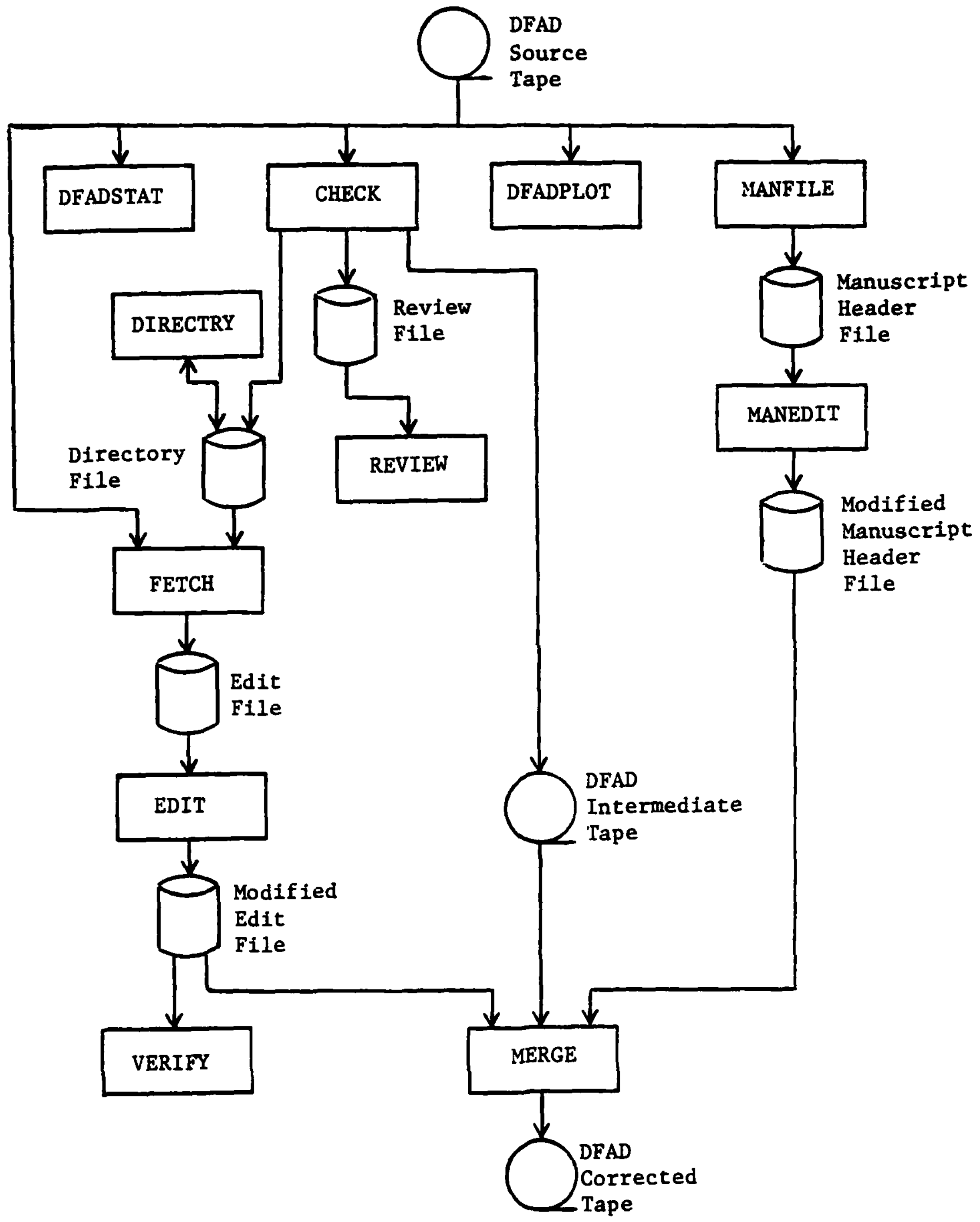


Fig 1. DFADEDIT Software Configuration

of a number of additional discrepancies. It was also apparent that a number of the more common and bothersome errors could often be corrected automatically.

Table I lists the tests performed by the DFADEDIT software. All errors listed are detected by the software. Those errors marked with a single asterisk can sometimes be corrected automatically by the software, and those flagged with two asterisks are always corrected automatically. The user should keep in mind that known errors of other types can often be corrected using editing capabilities inherent in the DFADEDIT software, even though the particular error condition may not be addressed in the table.

TABLE I

DFAD Errors Detected/Corrected

MANUSCRIPT HEADER ERRORS
- Illegal Manuscript Type
- Illegal Manuscript Level
- Illegal WAG (WAC) Number
- Illegal WAG (WAC) Cell
- Illegal WAG Cell
- Illegal Manuscript Latitude
- Illegal Manuscript Longitude
- Illegal Estimated Maximum Latitude
- Illegal Estimated Maximum Longitude

FAC 1 ERRORS
- Skewed Manuscript
- First Feature Not FAC 1
- First Feature Not Type Areal
- First Feature With Illegal Surface Material
- First Feature With Other Than Four Points

GENERAL FEATURE HEADER ERRORS
- * Illegal FAC
- Illegal Height
- Illegal Feature Identification Code (FIC)
- Illegal Surface Material Category (SMC)

POINT FEATURE ERRORS
- Illegal Orientation
- Illegal Length
- Illegal Width
- Illegal Number of Coordinates

LINEAR FEATURE ERRORS
- Illegal Directivity
- Illegal Width
- Illegal Number of Coordinates

AREAL FEATURE ERRORS
- Illegal Structures Code
- Illegal Tree Cover
- Illegal Roof Cover
- Illegal Number of Coordinates
- ** Open Areal Feature
- Areal Feature That Cannot Be Closed
- Clockwise Digitized Areal Feature
- Areal Feature With Zero Area

COORDINATE/DIGITIZATION ERRORS
 * Self-Intersecting Linear or Areal Features
   Points Outside Manuscript Boundary
 * Improperly Flagged Manuscript Exit Points
** Stack Points
** Spike Points

WARNINGS
   Average Length Spikes
   Potential Exit Point Errors on Skewed Manuscripts
   Point Features With Excessive Points
   Linear Features With Excessive Points
   Closed Linear Features
   Wrap-Around Features

2.3 Error Correction Procedures

This section provides an overview of all of the procedures involved in an error correction session. The process flow is depicted in the configuration diagram provided in Figure 1. Detailed discussion concerning specific program function and operation are contained in chapters 3-13 of this document.

The DFADEDIT software package is designed as a one-pass process. In other words, all errors are detected and corrected in one pass through the software if the programs are run in the proper sequence, in accordance with the guidance provided in this document.

2.3.1 Logon Procedures

After the user logs onto the system and receives the VAX/VMS "$" prompt, it is necessary to change the terminal speed to allow for dependable plot data transmission to the

HP-2649C terminal. To accomplish this, the user must enter the command "SET TERMINAL/SPEED=4800". After this command is issued, the user must physically set the terminal speed (via rotary switch) to 4800. The terminal is now ready for use by DFADEDIT programs. The user must reset the switch to 9600 after logging off.

2.3.2 Program DFADSTAT

Generally the first step to be performed with any DFAD tape to be corrected is to generate statistics on the contents of the tape using DFADSTAT. This program lists various types of summaries on the contents of the tape by manuscript, feature type, surface material category, feature identification code, etc. It provides information on the feature density and complexity for each manuscript, which can be used to estimate the time required for the error-correction process. DFADSTAT also lists manuscript reference and corner points, which may be inspected to detect manuscript gaps/overlaps.

2.3.3 Program DFADPLOT

DFADPLOT may be used to plot the entire contents or a specified subset of the features on a given manuscript on the user's graphics terminal. This graphical information may be cluttered and difficult to interpret on manuscripts with many features, but it can be helpful in deciding how to correct manuscript gaps/overlaps.

2.3.4 Program CHECK

CHECK is the first program in the actual error correction sequence, and at this point the user must decide whether to employ the automated correction capabilities. Once the user has made this decision and has provided the appropriate control parameters, CHECK processes user-specified manuscripts, scanning each manuscript header and feature for each of the error types listed in Table I. Whenever CHECK finds an error, it documents the error on the listing file and makes an entry in a directory file for the feature which contained the error. CHECK assigns an error severity code based upon the most severe error found within the feature. If the user has selected automated correction, CHECK attempts to correct the error automatically. If this attempt is successful, CHECK outputs both the original and the corrected versions of the feature to a review file, which may be examined by the user later to insure that all automated corrections are acceptable. CHECK fully documents all errors and corrections for the entire run. If CHECK performs automated corrections, it also generates an output tape which contains the same features as the input tape. The difference is that features on the output tape will have all stack and spike points removed, all open areal features closed, and all automated corrections implemented.

2.3.5 Program MANFILE

If CHECK discovers any discrepancies in any manuscript header on the input tape, the user may wish to use MANFILE to generate a manuscript header file. This file contains all the manuscript header data for each manuscript on the input tape. The file can be edited using MANEDIT.

2.3.6 Program MANEDIT

Using the MANEDIT program, the user has the capability to change any manuscript header parameter. MANEDIT also checks each change to verify that the new value lies within the range stated in the specification. As the user changes the manuscript header parameters, MANEDIT generates a corrected manuscript header file which will later be input to MERGE to generate a corrected output DFAD tape.

2.3.7 Program REVIEW

If automated correction is selected during CHECK processing, the user must review the automated corrections performed by CHECK using the REVIEW program. (If automated correction was not specified, REVIEW has no function in the correction of the current DFAD tape.) A review of the automated corrections is required because, for example, even though CHECK is able to resolve all intersections, a feature may be unacceptably distorted in the process. Using the analytical and graphical capabilities of REVIEW, the user can compare the original and automatically corrected versions of each feature. The discussion of the DIRECTRY program provides guidance on how to override unacceptable automated corrections.

2.3.8 Program DIRECTRY

The DIRECTRY program may be used to list or print the contents of the directory file. This listing provides a brief summary of the errors encountered by CHECK, along with some feature header information and the assigned error severity. A more important function of DIRECTRY involves editing the directory file, which in most cases is used as an input to the FETCH program to generate the edit file. DIRECTRY provides the user with the ability to explicitly include features with unacceptable automated corrections in the edit file. Automatically corrected features are normally ignored by FETCH in creating the edit file, since their assigned error severity is too low (they were, after all, corrected automatically). However, the user can flag the directory file entry for the offending feature so that it is placed into the edit file regardless of it's assigned error severity, in effect overriding CHECK's designation. Once the feature is in the edit file, it can be corrected interactively, and it will ultimately replace the unacceptable automatically corrected feature on the intermediate DFAD tape during MERGE processing. Finally, another DIRECTRY capability allows the user to create a directory file from scratch, which may then be used as input to FETCH to generate an edit file of user-selected features.

2.3.9 Program FETCH

Program FETCH uses the directory file to create the edit file. FETCH selects all directory file features with an error severity greater than or equal to a user-specified threshold to be included in the edit file. The resulting edit file contains all features which require interactive error correction. FETCH may also be used to examine features interactively from the input DFAD tape, and can plot or dump features in this mode.

2.3.10 Program EDIT

Once the edit file has been created, the next step involves interactive error analysis and correction. EDIT can modify existing edit file features by changing, inserting, deleting, or moving points, or it can create features from scratch or insert them from other existing edit files. The user sequences through the edit file and modifies features as required until the errors from each feature have been resolved. The output of EDIT is a new edit file with corrected features which are ready for verification.

2.3.11 Program VERIFY

VERIFY is used to scan the features in a corrected edit file to search for any errors that may have been overlooked in the interactive edit process. VERIFY performs basically the same tests as CHECK. Like CHECK, VERIFY also has automated error correction capability. Thus if a complex feature fails automated correction in CHECK due to one local flaw, the user may decide to correct that one error using EDIT, and then allow VERIFY to finish the error correction automatically. Like CHECK, VERIFY documents all errors, and if automated correction is selected, VERIFY also generates a review file and an output edit file.

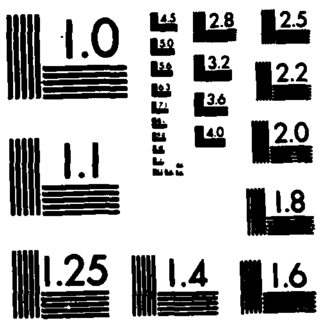
2.3.12 Program MERGE

The final step in the error correction process is MERGE. MERGE is used to assemble a corrected manuscript header file, a corrected edit file, or both with the remainder of the data from a DFAD tape. The tape input to MERGE is the intermediate DFAD tape if automated correction was specified in CHECK. If automated correction was not selected, the input tape for MERGE is the original DFAD tape. The output of MERGE is a corrected tape in standard DFAD format, and thus can be used as a replacement for the original. MERGE also produces a listing that indicates which features were substituted or added from the edit file and which manuscript headers were changed. The listing

AD-A124 893 IMPROVING THE USEABILITY OF THE DEFENSE-MAPPING AGENCY DIGITAL FEATURE AN..(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. M R NICOL

UNCLASSIFIED SEP 82 AFIT/GCS/EE/82S-9 F/G 9/2 NL

2/3



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

provides a general summary for each manuscript on the tape, and one for the entire tape.

At this point, the error correction process is complete. However, the user may wish to rerun statistics on the corrected tape. It is also possible, although not necessary, to process the corrected DFAD tape with CHECK to insure that no errors remain.

2.4 Internal Data Files

A number of disk-resident data files are created and used by the DFADEDIT software during the error correction process. These include directory files, manuscript header files, edit files, and review files. In addition, an intermediate DFAD tape is created by CHECK when used in automated mode. Each of these files is discussed separately below.

2.4.1 File Naming Conventions

Before discussing the actual internal data files, it is necessary to explain the file naming conventions used by the DFADEDIT software. All DFADEDIT programs generate the VAX/VMS system filename from a user-supplied root filename. This user-supplied root is an alphanumeric string of one to eight characters in length, made up of any combination of letters A-Z and numerals 0-9. A file type is appended to this user root to generate the system filename. The file type is ".MAN", ".DIR", ".REV", ".EDT", or ".LIS", depending upon whether the file is a manuscript header file, directory file, review file, edit file, or listing file, respectively. In addition, if the file is a listing file, the root is prefixed by "H" if the listing is generated by MANFILE or MANEDIT. If the file is a listing file generated by CHECK, REVIEW, DIRECTRY, FETCH, EDIT, VERIFY, or MERGE, the prefix is "C", "R", "D", "F", "E", "V", or "M", respectively.

2.4.2 File Headers

Each internal data file, regardless of type, has a file header record which contains information concerning the type of file (edit, review, etc.), the name of the DFADEDIT program that created the file and when, and a descriptive comment. This file header data is usually listed at the user's terminal or dumped to the listing file whenever one of these internal data files is opened for program access.

2.4.3 Manuscript Header File

The manuscript header file is created only when the user decides that one or more parameters in a manuscript header must be changed. The manuscript header file is initially created by MANFILE using the DFAD tape as input. The file contains an entry for each manuscript on the tape, which includes both the original and changed versions of each manuscript header. MANEDIT is used to change the desired manuscript header parameters, and creates a new manuscript header file containing the modified header information. This new header file may be edited again using MANEDIT, or it may be used directly as an input to MERGE in creating the corrected DFAD tape.

2.4.4 Directory File

The primary use of the directory file is as input to the FETCH program when creating an edit file. A directory file is created every time CHECK is run, and it contains an entry for each feature with an error severity other than zero. Each entry includes the feature header data, error severity, and counts of various errors detected. FETCH creates the edit file based upon the manuscript and FAC numbers of features, and their associated error codes, contained in the directory file. Directory files may also be created interactively by DIRECTRY from manuscript and FAC numbers supplied by the user. Directory files created in this manner contain error severity and error counts of zero, but still may be used as input to FETCH to create an edit file. Any directory file (regardless of which program created it) may be edited by DIRECTRY in order to specify that certain features are to be included or excluded from the edit file regardless of their error severity. This allows the user to override unacceptable automated corrections, and include any desired features for interactive editing.

2.4.5 Review File

The review file contains both the original and corrected coordinate information for features which CHECK or VERIFY have corrected automatically. Using the REVIEW program, the user may compare the original and corrected versions of the feature to be sure that the automated corrections are acceptable. This may be done by plotting all or affected parts of the feature on the users graphics terminal. If the user determines that the automated correction is acceptable, no further action need be taken, since the corrected feature has been written to the intermediate DFAD tape (CHECK), or the output edit file (VERIFY), where it will later be properly placed on the corrected output tape during the merge process. If on the other hand, the user decides that an automated correction from CHECK is unacceptable, he must then use DIRECTRY to mark the appropriate directory file entry so that the feature is included in the edit file regardless of severity level. If the review file was generated by VERIFY and unacceptable corrections are found, the output edit file from VERIFY should be deleted, and the subject error should be corrected interactively.

2.4.6 Edit File

The edit file may be created by FETCH, EDIT, or VERIFY. It contains header, coordinate, and error count data for all features which require interactive inspection or correction due to the nature of the errors detected during CHECK processing. Edit files may also contain features which are created from scratch using EDIT, or dummy features which are created by EDIT to cause deletion of specified features from the original DFAD tape during the merge process. Edit files are initially created by FETCH, using the directory file from CHECK, or one created by the user with DIRECTRY, as input. EDIT and VERIFY may also create output edit files as they operate on their respective input edit files.

2.4.7 Intermediate DFAD Tape

The intermediate DFAD tape is created by CHECK when automated error correction is specified. It is a copy of the original DFAD tape with the following exceptions:

1. Stack and spike points have been removed.
2. Open areal features have been closed.
3. Features with exit point errors and intersections which have been successfully corrected by CHECK are present on the output tape in their automatically corrected forms.

III. Program DFADSTAT

3.1 Program Description

DFADSTAT is an interactive program which allows the user to generate statistics on the contents of a DFAD tape. This program may be used prior to the start of error correction to determine the number of manuscripts on the input tape, and to analyze the relative manuscript complexity as well as feature complexity within the manuscript. The user may also use the manuscript reference and corner point information provided by DFADSTAT to manually assure that manuscript boundaries join properly, and no gaps or overlaps exist. DFADSTAT information may be helpful in estimating the approximate time required to process the entire tape or a given subset of manuscripts through the various phases of error correction. The program may also be run on the corrected tape to document the differences after error correction is complete.

DFADSTAT prompts the user for required input and control parameters. The user need only supply the input tape unit (A or B) and a comment line, and the program will then run to completion.

DFADSTAT processes a tape manuscript by manuscript, accumulating feature totals by feature type, feature identification code (FIC), surface material category (SMC), and general feature category. Summaries are generated by manuscript and for the entire run. The program terminates after reading the entire tape and generating all statistics.

3.2 Program Output

DFADSTAT generates an output listing which contains all the results of the statistics run. The output listing is written to the file "DFADSTAT.LIS". The first section of the listing is a manuscript summary which for each manuscript lists the boundaries in latitude and longitude, manuscript level, FAC 1 feature identification code, and counts of point, linear, and areal features contained on the manuscript. The second section of the listing contains a breakdown of the relative feature complexity by feature type for each manuscript. The third section contains a list of manuscript reference and corner points for each manuscript. The fourth section contains general summaries for the entire tape of features in various general feature identification code categories and surface material code categories. The fifth and final section of the listing contains a summary line for each feature identification code found on the tape, listing the total number of occurrences by feature type.

3.3 Program Operating Instructions

The following is an example of an interactive session in which DFADSTAT is executed. Computer prompts are underlined.

```
$ MOU/FOR MTA0:
$ DFADSTAT
DFADEDIT STATISTICS PROGRAM...
Enter Tape Unit (A/B):  A
Enter Comment:  TAPE S-6165, SOURCE
```

The program processes the tape and writes a summary line to the terminal for each manuscript. A summary for the tape by feature type is also written to the terminal once the entire tape has been processed. When processing is completed, the tape should be dismounted with the VAX/VMS command "DIS MTA0:".

IV. Program DFADPLOT

4.1 Program Description

The DFADPLOT program allows the user to plot all or selected subsets of the features from a specified manuscript on the input DFAD tape. The data is plotted on the user's graphics terminal. Although the plot may be of limited value due to the low resolution of the graphics terminal and relatively dense content for some manuscripts, it sometimes is useful when making decisions concerning the correction of manuscript gaps or overlaps, to visually examine the entire manuscript. Any combination of point, linear, and areal features may be plotted. A FIC range may be specified, allowing the user to be selective about which features are plotted.

DFADPLOT is an interactive program which operates on a DFAD input tape and user commands. The program prompts for the input tape device and the tape number, and then requests parameters to control the content of the plot to be generated. The user is asked to provide an optional FIC range to plot from, and to signify whether he wishes to plot point, linear, or areal features, or some combination.

Finally, the user is prompted for a manuscript number. DFADPLOT then skips to the appropriate manuscript and plots the specified features.

The plot generated by DFADPLOT is bounded by a dashed line which represents the manuscript boundary. All features are plotted within the dashed boundary in the order they are encountered on the input tape. The plot is labeled with the tape number, manuscript number, latitude/longitude extents, FIC range, and the number of each type of feature plotted. When a plot is completed, the program asks if more plots are to be generated. If so, the process is repeated, starting with the FIC range prompt. If no more plots are desired, the program terminates. DFADPLOT generates no listed output.

3.2 Program Operating Instructions

The following is an example of a DFADPLOT session:

```
$ MOU/FOR MTBO:
$ DFADPLOT
DFADEDIT MANUSCRIPT PLOT...
Enter Tape Number:  TAPE S-6165
Enter Tape Unit (A/B):  B
Do You Wish to Plot From a FIC Range?  Y
Enter FIC Range (FIC1,FIC2):  800,999
Enter Feature Code:
1 = Point
2 = Linear
3 = Point and Linear
4 = Areal
5 = Point and Areal
6 = Linear and Areal
7 = Point, Linear, and Areal:  3
Enter Manuscript Number:  12
```

In this example, the program proceeds to plot point and linear features in the FIC range of 800 through 999 from manuscript 12 of the tape mounted on tape drive MTB0:. When the requested plot is complete, the program will prompt:

<u>More Plots Desired?</u>

If the user answers yes (Y), The program will loop back to the request for FIC range for the next plot. If the user answers no (N), the program will terminate. The user should then dismount the tape using the command "DIS MTB0:".

V. Program CHECK

5.1 Program Description

CHECK is the first program employed in the actual error correction process. It processes user-specified manuscripts from an input DFAD tape and performs tests on manuscript header parameters, culture feature header data, and culture feature coordinate data. If errors are detected, CHECK attempts automatic correction and/or it provides information to allow the user to correct errors interactively. The program is designed to provide maximum control and traceability over the error correction process. In addition to a number of user parameters which control automated correction, a capablilty is provided to review any features which are automatically corrected to assure that no unacceptable automated corrections pass undetected through the error correction process. All errors detected or corrected are fully documented.

The first function of CHECK is to accept the user options for the run (see section 5.4). Once these parameters are defined, CHECK begins processing the input tape, attempting automated correction of some types of errors if so requested. If any of the user parameters which control automated error correction are exceeded, the feature error involved is said to be "fatal" to the automated correction process for that feature. Once a fatal error is encountered, processing continues in error "detection-only" mode, and no more automated correction is attempted for that feature. All errors are well documented to facilitate interactive feature correction. The control parameters and their effect on error correction are discussed in the following paragraphs.

5.1.1 Manuscript Header Processing

CHECK performs range tests on parameters in the manuscript header as defined in Table II below. Although no error severity is associated with discrepancies detected in the manuscript header data, an error is logged in the output listing. If desired, the manuscript header data may be edited using programs MANFILE and MANEDIT. In any case, all manuscript header data is listed on the output listing along with an approximate area of coverage for the manuscript.

TABLE II

Manuscript Header Permissible Values

Parameter	Permissible Range
*********	*****************
Type	1 or 2
Level	0 or 7
WAG (WAC) Number	1 to 9999
WAG (WAC) Cell	1 to 25
WAG Cell	1 to 16
Origin Latitude	-3240000 to 3240000
Origin Longitude	-6480000 to 6480000
Est. Max Latitude	1 to 360000
Est. Max Longitude	1 to 360000

5.1.2 Feature Processing

The following sections describe the processing applied to each feature on the DFAD input tape.

5.1.2.1 Error Severity

As CHECK processes selected manuscripts from the DFAD input tape, it classifies all feature discrepancies with an error severity code. These error severity codes are defined in Table III. Features with errors receive a severity code of 1 through 6, where 6 is the most severe error. A severity code of 0 is assigned to features with no errors of the types addressed by this software (see Table I).

TABLE III

Error Severity Codes

Severity ********	Meaning *******
0	No errors or warnings
1	Minor errors - corrected automatically
2	Exit point errors - corrected automatically
3	Intersection errors - corrected automatically
4	Warning - no corrective action taken
5	Fatal error - automated correction failed
6	Fatal error - no automated correction attempted

Error severity 1 is applied to features with stack points or spike points, or to open areal features. If complications arise during the automated correction of any severity 1 errors, the automated correction is aborted and the severity is raised to 5.

Error severity 2 is applied to features with exit point errors which can be corrected automatically. These are usually points that geographically (by latitude and longitude) are manuscript exit points, but are not flagged as such with a negative latitude. Fatal exit point errors, such as points which are flagged but do not lie on the manuscript boundary are also detected, but they are not corrected and are assigned a severity of 5.

Error severity 3 is used for features which have various types of intersections which can automatically be resolved by controlled deletion or rearrangement of a number of points within the feature. Any deletion or rearrangement is done subject to user control parameters in order to preserve the fidelity of the original feature.

Error severity 4 is used to warn of features with possible errors. Examples include linear features with more points than a user-specified maximum, "closed" linear features, features with average length spikes, or features on skewed manuscripts which have potential exit point errors. No automated correction is attempted on such features.

Error severity 5 is applied to features which initially have a severity of 1, 2, or 3, but subsequently fail automated correction attempts. A number of user parameters are available to control automated corrections, and any automated attempt which violates any of these controls may cause such a failure. The controls include a limit on the number of automatically correctable intersections and exit point errors within a feature, and limits to control the variation in the area of areal features.

Error severity 6 is used when features have errors which can only be corrected interactively. These include manuscript or feature header parameters out of range, feature points outside the manuscript boundary, some types of exit point errors, clockwise areal features, areal features with zero area, and the more severe intersection errors.

5.1.2.2 Feature Error Tests

The following discussion provides details on error testing as applied to each feature. If the feature is the first feature on the manuscript, the feature header is checked to insure that the FAC is 1, the type is 2 (areal), the surface material category (SMC) is 6 or 10, and the number of points is four. The coordinates are tested for rectangularity, and any deviation causes the manuscript to be reported as skewed. Failure of any of these background feature tests is a fatal error (severity 6).

Processing continues as each feature is checked in the order in which it is encountered on the manuscript. General checks are performed on the feature header regardless of feature type. Each feature must have a FAC which is greater than that of the preceeding feature. Each feature must also have a valid feature identification code (FIC), surface material category, and height. Specific feature parameters are also tested according to feature type. A severity 6 error is issued if of any these conditions are not met. The feature header parameters are tested for valid range according to the values defined in Table IV below.

TABLE IV

Feature Header Permissible Values

Parameter	Permissible Values
General	
FAC	1 to 16383
Height	-511 to 511
FIC	Valid FIC
SMC	1 to 13
Point Feature	
Orientation	0 to 31 or 63
Length	0 to 127
Width	0 to 127
Number of Points	1 to 2047
Linear Feature	
Directivity	1 to 3
Width	0 to 127
Number of Points	2 to 8191
Areal Feature	
Structures	0 to 15
Tree Cover	0 to 10
Roof Cover	0 to 10
Number of Points	4 to 8191

Next, linear features are checked for closure, and a warning (severity 4) is logged if the last point of the feature has the same geographic coordinates as the first. An entry is made in the directory file so that the feature can be optionally included in the edit file and visually inspected using EDIT.

If the feature is an areal feature and if the last point of the feature is not geographically equal to the first point, the feature is an "open areal". Open areal features are closed automatically by CHECK. This is accomplished by adding an additional point to the feature with delta latitude and longitude identical to the first point. This is a severity 1 error.

Next, regardless of feature type, all stack points and spike points are located and removed. This operation is performed automatically whether or not the automated correction option is selected, since stack and spike points are relatively low-risk (severity 1) errors.

The number of points for each point feature is compared to a user-specified parameter (PFPNTWRN), and a warning is issued if the parameter is exceeded. The user controls a parameter (LFPNTWRN) used in a similar test for linear features.

The next check is called an average length test, since it compares the distance between a point and its neighboring points with the average length computed for all line segments of the feature. This test is applied only to linear and areal features. If any point is offset from its neighbors by a user-specified multiple (AVGLFCTR) of the computed average, and if the two neighboring points are within a user-specified distance of one another (AVGLDIST), a warning (severity 4) reporting a possible average length spike at this location is logged.

This is followed by a check for any feature coordinate which lies outside the manuscript boundary as defined by the coordinates of the FAC 1 feature. This test is performed on all features regardless of type. If the manuscript is non-rectangular (skewed), each feature coordinate is tested against the smallest rectangular area defined by the FAC 1 feature. Any point which falls outside the boundary triggers a fatal error (severity 6).

If the automated correction option (AUTOMODE) has been selected, and if intersection detection (INTERSCT) has been specified, the next step involves automated detection/correction of any feature self-intersections. The automated correction process is documented separately in section 5.3. If CHECK can resolve all intersections it encounters in a feature, the feature is output to the review file. In this way, automated corrections may be reviewed by the user when the CHECK run is completed, and he may take remedial action if the automated correction is determined to be unacceptable.

If the automated correction algorithm encounters an intersection or set of intersections that is for any reason fatal to the automated correction process, an error is logged and processing continues by restoring the feature to its original form and rescanning for intersections with no attempt at correction. All detected errors are documented to aid in manual correction with the EDIT program.

Once intersection detection/correction is complete, the next step is to check for properly flagged manuscript exit points. Exit point errors are corrected automatically unless CHECK encounters a point that is flagged as an exit point but is not on the manuscript boundary, or unless the feature is contained in a skewed manuscript.

If a skewed manuscript boundary was detected, all linear and areal features will be checked to determine if they may contain exit point errors once the skew is corrected. Such features are flagged with error severity 4.

When CHECK has completed all tests on all features of a manuscript, a summary for that manuscript is output to the listing. When the last selected manuscript or the last manuscript on the tape has been processed, a summary for the entire run is written to the listing file. The program then closes all files, rewinds the tapes, and terminates.

5.2 Operating Environment

5.2.1 Modes of Operation

CHECK may be run in interactive or batch mode. CHECK operates most efficiently in batch mode since there are no user decision points once execution has been initiated. However, interactive mode is available to allow the user to monitor the progress of the error correction process if so desired. In interactive mode, the user is prompted for input parameters and options, a message is output to the user's terminal whenever any error is found, and manuscript summaries are reported. In batch mode, the user provides inputs and options via a command file, and no interaction takes place.

5.2.2 Required Peripherals

CHECK requires one tape drive for the input DFAD tape, along with sufficient disk capacity for the creation and generation of the directory, review, and listing files as required. If automated correction is used, CHECK requires an additional tape drive for the intermediate DFAD output tape.

5.2.3 Processing Time

Although exact processing time for a manuscript cannot be predicted, the following general guidelines apply. Intersection detection/correction is by far the most time-consuming of the error tests. If this option (INTERSCT) is selected, processing time ranges from fractions of a second for features with one point to 20-40 minutes for features with 7000 or more points. If intersections are ignored, processing time ranges from fractions of a second to 5-10 seconds per feature. Thus the time to process a manuscript for intersections is highly dependent upon the number of features contained in the manuscript and their relative complexity in terms of numbers of points. Another complication is introduced by the processing load on the system due to other users. However, in the absence of other users, an estimated processing time of 15-30 minutes per manuscript, on the average, may be used. Remember that this is an average number, and it is subject to wide variations in either direction for individual manuscripts.

5.3 Automated Corrections

Although the frequency and character of some types of errors which occur in the DFAD data are unpredictable, there are several classes of errors or anomalies which are exact or which always occur in one of a finite number of ways. The errors which exhibit these characteristics are also some of the most frequently occurring errors and are thus most bothersome to users of the data. Examples of errors which fall into this category include stack and spike points, open areal features, most exit point errors, and a high percentage of intersections. These errors have thus received the emphasis of the development of automated correction techniques.

The occurrence of stack and spike points is quite common in the data, and fortunately the correction of these two types of errors is straightforward. Thus, stack and spike points are detected and removed automatically by the software whether or not automated correction is selected.

The correction (closing) of open areal features is also a trivial problem, and for that reason this error is always handled automatically by the DFADEDIT software.

The other classes of errors which lend themselves to automated correction are self-intersecting line segments in linear and areal features, and exit point errors. These are discussed separately below.

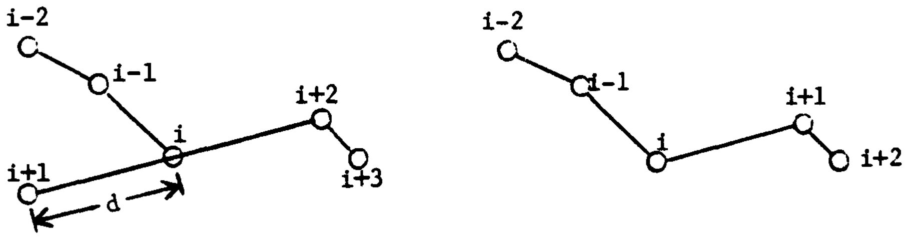
5.3.1 Intersections

An intersection in a linear or areal feature is characterized by any two non-consecutive line segments which touch or cross, or by any two consecutive line segments which touch at some point other than their common endpoint. The intersection detection algorithm must thus compare every feature line segment with every other line segment in the feature. It is easy to understand how performing this test on very large features (i.e., features with 8000 points) can involve excessive processing time. However, it is possible to correct the intersection automatically in about the same amount of time required to simply detect the error. Thus automated correction of intersections saves considerable time and relieves the user of tedious work.

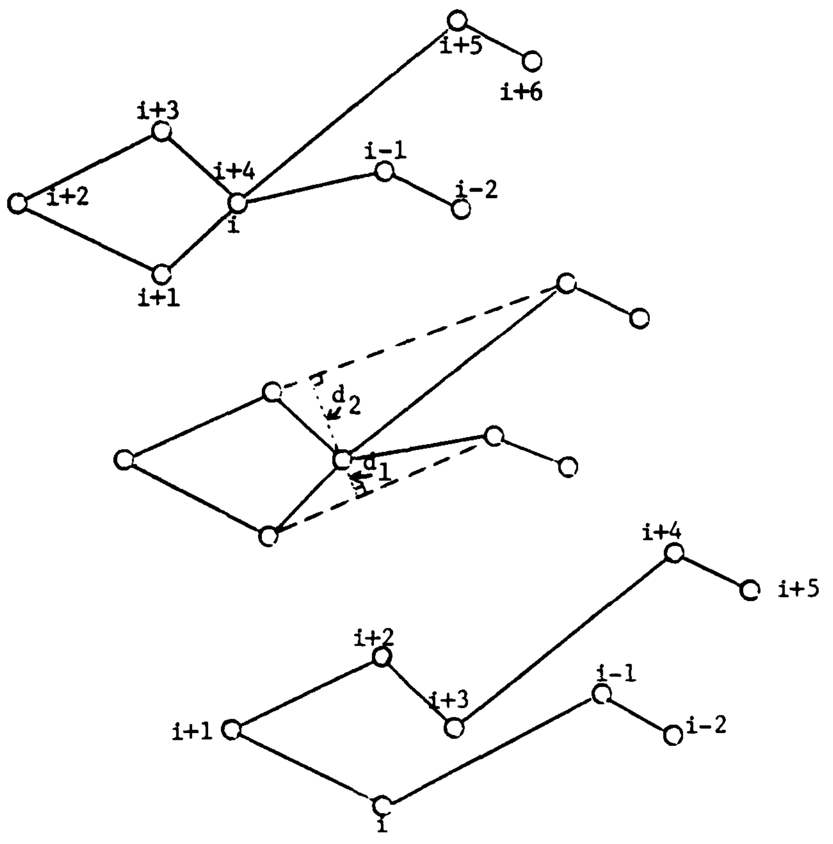
There are two basic approaches used in the automated correction of an intersection. One method involves careful deletion of a feature point, and the other involves "unwinding" the intersection by changing the order of some of the feature coordinates in the area of the intersection, without deleting points. Each of these methods is discussed below.

Point deletion is a correction technique which is applied to the types of intersections shown in Figure 2. It is necessary to delete a point because the definition of the feature is such that there is no other obvious method to resolve the intersection. All point deletion is performed under control of the user parameter DELETDST. Thus, for the types of intersections shown in Figure 2, if the distance "d" is in any case greater than DELETDST, the automated correction attempt is aborted and the feature is set aside for interactive correction. The automated correction also fails if any attempt is made to delete a point flagged as a manuscript exit point.

The "unwind" method of intersection correction is used to resolve what are known as general intersections. General intersections are characterized by two line segments which simply cross each other at some point other than their endpoints. Examples of general intersections are shown in Figure 3. Note that by rearranging the feature points as shown, the intersection can be resolved while in most cases preserving the original local character of the feature. The "unwind" algorithm for intersection correction is subject to control by the user parameter UNWINDST. If the distance "d" shown in Figure 3 in any case exceeds UNWINDST, the



a. Lines (i,i+1) and (i+1,i+2) are coincident



b. Points i,i+4 are coincident

Fig 2. Point Deletion Intersection Correction

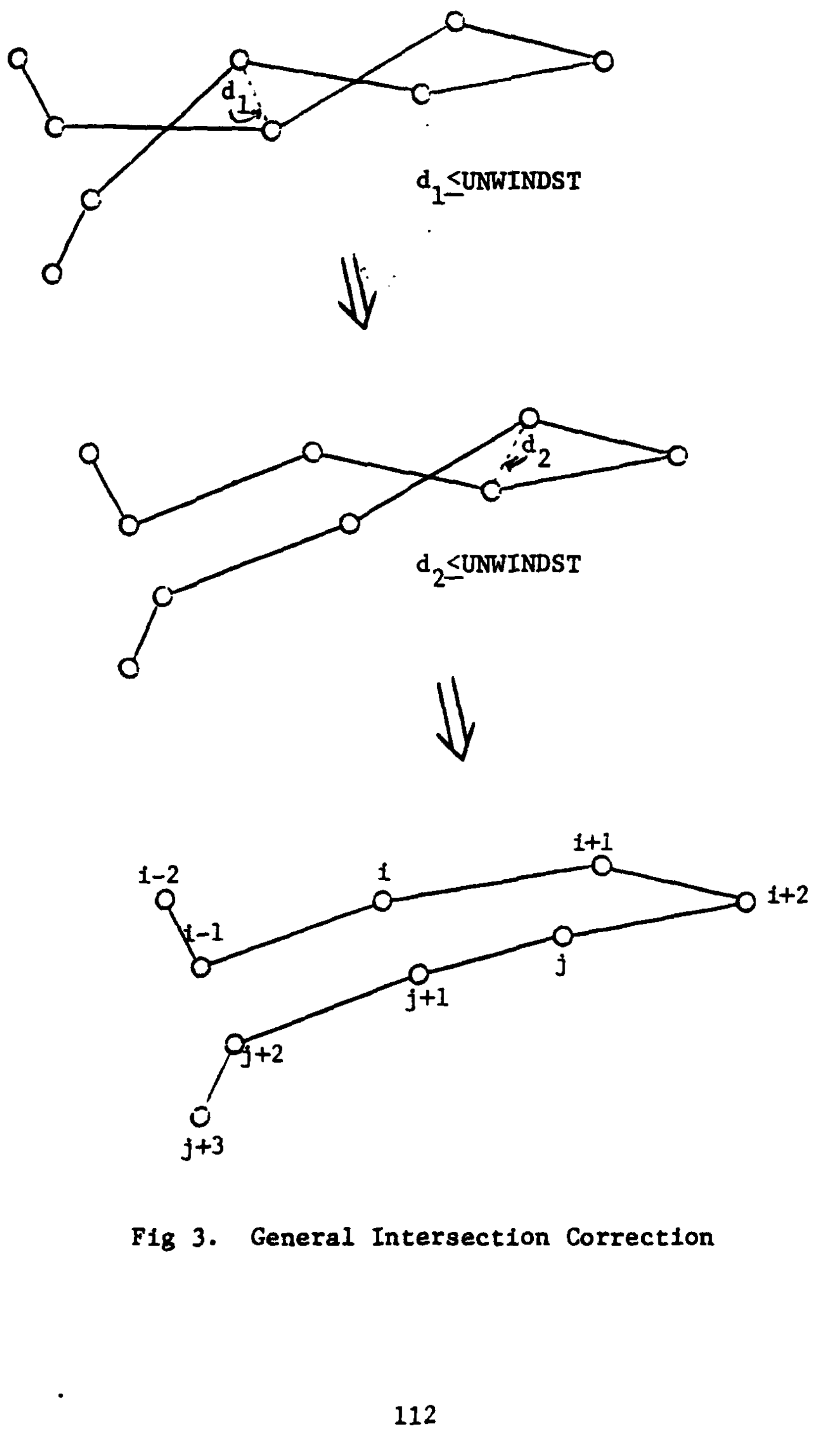


Fig 3. General Intersection Correction

automated correction is aborted, and the feature must be corrected interactively. In any case, each delete or unwind operation is documented in the listing file.

5.3.2 Exit Point Errors

Exit point errors in most cases can be corrected automatically. This is due to the fact that most exit point errors involve points which by their location are indeed manuscript exit points, but are not flagged as such. It is a simple operation to have CHECK flag the point as an exit point. However, if a point is flagged as an exit point, but is not a legal exit point due to its location or for some other reason, the error is flagged at severity 6, and is left for the user to correct interactively. All exit point errors and corrections are also documented in the listing file.

5.4 User Input and Control Parameters

The user provides a number of inputs to the CHECK program, the first of which is the DFAD tape to be scanned or corrected. In addition, the user supplies a user file name, which is used to generate the system file names for the directory, review, and listing files. The user also provides a comment line which describes the tape to be processed, or the processing selected. This comment appears on the listing and is carried along in all the internal files throughout the error correction process. Finally the user supplies a number of user control parameters to select various error correction options. These parameters are defined in Table V below. The parameters are specified to CHECK in the form "PARAMETER=value". For details on entering the parameters, see section 5.6.

TABLE V

CHECK User Control Parameters

Parameter *********	Description ***********
AFAREVAR	AREAL FEATURE AREA VARIATION This parameter is used in conjunction with automated intersection correction, and is a percentage of variation (increase or decrease) allowed during automated intersection correction for areal features. After each automated correction, the new feature area is computed, and if the area falls outside tha allowable limits, a fatal error (severity 5) is issued. The default is 10 percent, and permissible values range from 0 to 50.
AFPNTVAR	AREAL FEATURE POINT VARIATION This parameter is used in conjunction with automated intersection correction, and is a percentage of variation allowed in the number of points for areal features undergoing automated correction. If the number of feature points changes more than the limit, a fatal error (severity 5) is issued. The default is 10, and the allowable range is 0 to 50.
AUTOMAX	MAXIMUM AUTOMATED CORRECTIONS (PER FEATURE) This parameter specifies the maximum number of automated corrections (exit points and intersections) allowed per feature. If automated correction attempts to exceed this number, the feature is assigned error severity 5, and automated correction is disabled for the current feature. The maximum allowable value is the default value of 200.
AUTOMODE	AUTOMATED CORRECTION SWITCH This parameter enables automated correction if true, and disables automated correction if false. Options are T for true and F for false. The default value is true.

AVGLDIST	AVERAGE LENGTH DISTANCE This parameter, along with AVGLFCTR, controls the average length test used to detect spikes. AVGLDIST is the allowed maximum separation between the two points which are adjacent to the test point. The default value is 10, and AVGLDIST is specified in tenths of arc-seconds. For details, refer to the discussion of the average length test in section 5.1.2.2.
AVGLFCTR	AVERAGE LENGTH FACTOR This parameter, along with AVGLDIST, controls the average length test used to detect spikes. AVGLFCTR is an allowed multiple of the average segment length between any two adjacent feature points computed for each feature. The default value is 4. For details, refer to the discussion of the average length test in section 5.1.2.2.
DELETDST	DELETION DISTANCE LIMIT This parameter controls the deletion of points when such action is required by automated intersection correction. DELETDST is specified by the user in tenths of arc seconds and is the maximum allowable perpendicular distance from a point to a line that joins its two adjacent points. For more details, see the discussion in section 5.3. The default value is 25.
FIRSTMAN	FIRST MANUSCRIPT This parameter selects the first manuscript to be processed from the input tape. The default value is 1.
INTERSCT	INTERSECTION DETECTION/CORRECTION SWITCH This parameter enables intersection detection if true, and disables intersection detection if false. Options are T (true) and F (false). The default value is true. Both INTERSCT and AUTOMODE must be TRUE to perform full automated error correction.
LASTMAN	LAST MANUSCRIPT This parameter selects the last manuscript to be processed from the input tape. If the user supplied value is larger than the number of manuscripts on the tape, processing will terminate normally after the last manuscript on the tape has been processed. The default value is 9999.

LFPNTVAR	LINEAR FEATURE POINT VARIATION This parameter is used in conjunction with automated intersection correction, and is a percentage of variation allowed in the number of points for linear features undergoing automated correction. If the number of feature points falls below the limit, a fatal error (severity 5) is issued. The default is 10, and allowable values range from 0 to 50.
LFPNTWRN	LINEAR FEATURE POINT WARNING THRESHOLD This parameter specifies that CHECK will issue a warning (error severity 4) for any linear feature with more than LFPNTWRN points. The default value is 200.
PFPNTWRN	POINT FEATURE POINT WARNING THRESHOLD This parameter specifies that CHECK will issue a warning (error severity 4) for any point feature with more than PFPNTWRN points. The default value is 100.
REVWSEV	REVIEW FILE THRESHOLD (LOWER LIMIT) This parameter allows the user to control the contents of the review file generated by CHECK when AUTOMODE is true. The default value is 3, which means that only features with error severity 3 (automatically corrected intersections) are placed in the review file. The user may set REVWSEV equal to 2, in which case features with automatically corrected exit point errors will also be placed in the review file. These (2 and 3) are the only allowable values for REVWSEV.
TAPEIN	INPUT TAPE This parameter selects the input tape drive. Permissible values are A or B, and the default is A. If automated correction is enabled (AUTOMODE=T), the output tape is assigned to the remaining tape drive.
UNWINDST	UNWIND DISTANCE LIMIT This parameter controls the "unwind" algorithm used in the automated correction of some types of intersections. It is specified in tenths of arc seconds, and has a default value of 25. For more discussion, see section 5.3.

5.5 Program Output

This section discusses the various outputs from CHECK, including data files, interactive error messages, listing file contents, and the intermediate output tape.

5.5.1 Data Files

CHECK produces up to three data files which reside on system disks and are thus subsequently available for access by other programs during the error correction process. A directory file is produced by CHECK for every run. The review file is produced when the user parameter AUTOMODE is true. Finally, the listing file, like the directory file, is produced for each run of CHECK independent of user parameters.

5.5.2 Interactive Error Messages

When CHECK is run interactively, an error message is written to the user terminal for each error encountered. The message appears in the following format:

```
mmmm-Man    fffff-FAC    t-Type    pppp-Points    ee-Err
```

The first four parameters are the manuscript and FAC number, type, and number of points for the offending feature, respectively. The last parameter indicates the type of error discovered, and may take one of two forms. If "ee" is a number, it refers to one of the errors listed in Table VI below. If "ee" is of the form An (n=1 to 7), the message indicates that an error was detected and corrected automatically. (If n is 1, the error is an exit point; if n is 2-7, the error is some type of intersection). These messages are provided simply for user information, and the normal, more descriptive error message will be provided in the listing file. Fatal errors (those with severity greater than 4) imply that the error will terminate automated correction attempts for the current feature.

TABLE VI

Interactive Error Messages and Error Severity Codes

Code	Severity	Meaning
****	********	*******
1		Illegal Manuscript Type
2		Illegal Manuscript Level
3		Illegal WAG (WAC) Number
4		Illegal WAG (WAC) Cell
5		Illegal WAG Cell
6		Illegal Manuscript Latitude
7		Illegal Manuscript Longitude
8		Warning - Estimated Maximum Latitude
9		Warning - Estimated Maximum Longitude
10	6	Manuscript Boundary is Skewed
11	6	First Feature on Manuscript not FAC 1
12	6	FAC 1 has Illegal Type
13	6	FAC 1 has Illegal Surface Material Category
14	6	FAC 1 has Illegal Number of Points
15	6	Illegal FAC
16	6	Illegal Height
17	6	Illegal Feature Identification Code (FIC)
18	6	Illegal Surface Material Category
19	6	Illegal Point Feature Orientation
20	6	Illegal Point Feature Length
21	6	Illegal Point Feature Width
22	6	Point Feature with Illegal Number of Points
23	6	Illegal Linear Feature Directivity
24	6	Illegal Linear Feature Width
25	6	Linear Feature with Illegal Number of Points
26	6	Illegal Areal Feature Structures Code
27	6	Illegal Areal Feature Tree Cover
28	6	Illegal Areal Feature Roof Cover
29	6	Areal Feature with Illegal Number of Points
30	4	Warning - Closed Linear Feature
31	5	Open Areal Feature That Cannot be Closed
32	4	Warning - Average Length Spike
33	5/6	Exit Point Error
34	6	Point Outside Manuscript
35	6	Feature Digitized Clockwise
36	6	Feature Has Zero Area
37	5	Fatal Coincident Point Error
38	5	Fatal Closure Intersection
39	5	Fatal Coincident Line Error
40	5	Fatal Feature Compression Error
41	5	Fatal Point-Line Intersection
42	5	Fatal General Intersection
43	5	Fatal Automated Correction Overflow
44	6	Coincident Lines

45	6	Intersecting Lines
46	6	Coincident Points
47	1	Stacks, Spikes, or Areal Feature Closure
48	4	Warning - Potential Exit Point Error(s)
49	4	Warning - Point Feature Points
50	4	Warning - Linear Feature Points
51	5	Fatal Point Variation
52	5	Fatal Areal Feature Area Variation
53	4	Warning - Potential Wraparound Error

5.5.3 Error Log Listing

CHECK produces a listing file for each run. The listing contains manuscript header, feature, error, and CHECK processing summaries for each manuscript processed and for the entire run. These topics are discussed separately below.

The first page of the listing file contains the user supplied comment and the names of the directory and review files being produced by this run. All appropriate user control parameters are also listed.

All manuscript header parameters for each processed manuscript are written to the listing file, along with the manuscript corner points and the manuscript area of coverage. Any errors or warnings detected in the manuscript header are listed just below the manuscript header data.

Following the manuscript header data for each manuscript is an error report that contains one line for each error encountered in the manuscript. One feature may have numerous error lines. Each error line includes the manuscript and FAC number, a descriptive error message (similar to those listed in Table VI), several feature coordinate indices that have different meanings for each error, a distance (automated deletion or unwind distance) if applicable, the current error severity, feature header data, and for areal features, the current feature area. The coordinate indices indicate which feature points are in error. For example, an exit point error lists the point which is in error, while a general intersection lists the two indices which are the first points of the two intersecting line segments. An automatically corrected coincident point error lists three indices; the first and second are the points which are coincident, and the third indicates which point is deleted to resolve the error.

Finally, a summary is printed for each manuscript which lists the number of each type of feature on the manuscript, the number of manuscript header errors/warnings, the number of features at each error severity level, and the number of occurrences of the more common errors. Also included is some processing time information from CHECK. This same type of summary is provided at the end of the listing for the entire run.

5.6 Program Operating Instructions

CHECK may be used either in interactive or batch mode. Each is discussed below.

5.6.1 Interactive Processing

For short jobs, it may be desirable to run CHECK interactively. The following sequence of prompts/responses is an example of an interactive CHECK execution, which performs automated error correction of the first six manuscripts from an input tape on drive MTB0:. By default, the output tape is on the remaining drive, MTA0:.

```
$ MOU/FOR MTA0:
$ MOU/FOR MTB0:
$ CHECKI
DFADEDIT CHECK PROGRAM...
Enter Comment Line:  Tape S-6165, MANUSCRIPTS 1-6
Enter Filename 6165
       .
       .
 (Control Parameters are Listed at the Terminal)
       .
       .
Do You Wish To Change Control Parameters?  Y
Parameter?  LASTMAN=6
Parameter?  TAPEIN = B
Parameter?  END
       .
       .
 (Control Parameters are Listed at the Terminal)
       .
       .
Do You Wish To Change Control Parameters?  N
```

At this point, CHECK begins to process the desired manuscripts. When the task completes, the user should dismount the tape and print the listing with the command "PRI C6165.LIS", where 6165 is the filename entered in response to the filename prompt above.

5.6.2 Batch Processing

Batch is the preferred mode for CHECK. The following example shows a command file set up for batch execution of CHECK. Note that tape allocation and mounting is handled via the command file, and therefore the user should insure that the tapes are physically mounted on the proper drives and that the drives are on-line prior to submitting the command file to the batch job queue. The tape drive(s) being used must be free (unallocated to any user) so that the batch job can allocate them for itself. (The text in the example following the "!" are comments, and should not be included in the actual command file.)

```
$ MOU/FOR MTA0:                 !Mount output tape
$ MOU/FOR MTB0:                 !Mount input tape
$ CHECK                         !Run batch version of CHECK
TAPE S-6165, 1-6                !Comment line
6165                            !Filename
LASTMAN = 6                     !Set last manuscript
TAPEIN = B                      !Input tape on MTB0:
END                             !End of user parameters
$ DIS MTA0:                     !Dismount output tape
$ DIS MTB0:                     !Dismount input tape
$ PRI C6165 LIS                 !Print listing file
```

Program MANFILE

6.1 Program Description

If CHECK detects any errors in the manuscript header parameters, or if known errors exist in any manuscript header on the DFAD tape, MANFILE is used to generate a file containing all manuscript header data from the tape. This file can then be edited by the user with MANEDIT to modify the faulty header data. MANFILE and MANEDIT provide the only means to change parameters in the manuscript header.

MANFILE is an interactive program, however little user interaction is required. The program prompts for the input tape drive (A or B), a comment line, and a user filename for the manuscript header file. Once the user provides this data, MANFILE reads through all manuscripts on the input DFAD tape and stores two copies of each manuscript header in the manuscript header file. When the end-of-tape is reached, MANFILE terminates. A record of the manuscript headers processed is contained in the listing file named "Hroot.LIS", where "root" is the one to eight character user filename.

6.2 Program Operating Instructions

MANFILE is an interactive program executed in the following manner:

```
$ MOU/FOR MTA0:
$ MANFILE
DFADEDIT MANUSCRIPT HEADER FILE PROGRAM...
Enter Input Tape Unit (A/B):  A
Enter Comment:  TAPE S-6165
Enter Manuscript Header Filename:  6165
```

The program then reads the tape, issuing a summary at the terminal for each manuscript header encountered and writing all the parameters for each manuscript header to the file "6165.MAN". When the program finishes, the user should dismount the tape and print the listing from file "H6165.LIS".

VII. Program MANEDIT

7.1 Program Description

MANEDIT is used to examine or modify the parameters in a manuscript header file produced by MANFILE or by MANEDIT. MANEDIT is the only program in the DFADEDIT software package capable of changing parameters in the manuscript header. The output of MANEDIT is a modified manuscript header file which contains the user-corrected manuscript header data. This file can be used as input to the MERGE program when creating the corrected DFAD tape.

MANEDIT is an interactive program which allows the user to change any field in the manuscript header of any manuscript on the subject DFAD tape. This is accomplished via a number of interactive commands which the user employs to examine or change various parameter values, reset to original values, and check the changed parameters for compliance with the DFAD specification. The program begins execution by prompting the user for filenames for both the input and output manuscript header files. MANEDIT then sequences through the input manuscript header file manuscripts on user demand. The output file is generated as the user proceeds through the input file. If the user changes a header parameter, the changed value is stored with the second set of header parameters for the current manuscript. In this way, the original parameter values are always maintained and can be quickly restored if so desired. MANEDIT generates a listing file which documents all changes made to manuscript header parameters during the current run. The listing file is named "Hroot.LIS", where "root" is the one to eight character user filename provided for the output manuscript header file.

7.2 MANEDIT Commands

The user controls the operation of MANEDIT through the use of 17 interactive commands. Ten of these commands change fields in the manuscript header, and the other seven perform special functions such as checking for specification compliance, displaying available commands, skipping to desired manuscript header records, restoring original parameter values, etc. Only the first four characters of each command need be entered. Each of the commands is discussed in Table VII.

TABLE VII

MANEDIT Commands

Command *******	Function ********
CHECK	CHECK PARAMETERS This command checks all parameters of the current manuscript header for compliance with the specification, and prints an error at the user's terminal if any discrepancies are found.
EXIT	EXIT PROGRAM This command exits the program. Prior to exiting, MANEDIT copies any manuscript header data remaining in the input file to the output file.
HELP	DISPLAY COMMANDS This command displays a summary of all MANEDIT commands at the user's terminal.
LATMAX	CHANGE MAX. EST. LATITUDE This command changes the estimated maximum latitude parameter.
LATORG	CHANGE ORIGIN LATITUDE This command changes the manuscript origin latitude parameter.
LEVTYP	CHANGE LEVEL This command changes the manuscript level parameter.
LONMAX	CHANGE MAX. EST. LONGITUDE This command changes the estimated maximum longitude parameter.
LONORG	CHANGE ORIGIN LONGITUDE This command changes the manuscript origin longitude parameter.
MANTYP	CHANGE MANUSCRIPT TYPE This command changes the manuscript type parameter.

NEXT	GET NEXT MANUSCRIPT HEADER This command copies the current manuscript header data to the output manuscript header file, and retrieves the next header from the input file.
RESTORE	RESTORE ORIGINAL PARAMETERS This command restores the parameters for the current manuscript header to their original values.
SHOW	SHOW MANUSCRIPT PARAMETERS This command displays both the original and modified values of the current manuscript header at the user's terminal.
SKIP	SKIP TO DESIRED MANUSCRIPT This command skips to a specified manuscript header by copying all header records up to the one desired from the input file to the output file.
WACCEL	CHANGE WAG (WAC) CELL This command changes the WAG (WAC) Cell parameter.
WAGCEL	CHANGE WAG CELL This command changes the WAG Cell parameter.
WACNUM	CHANGE WAG (WAC) NUMBER This command changes the WAG (WAC) Number parameter.
ZERO	CHANGE ZEROES FIELD This command changes the Zeroes field of the manuscript header.

7.3 Program Operating Instructions

The following is an example of the commands required to interactively execute MANEDIT:

```
$ MANEDIT
DFADEDIT MANUSCRIPT HEADER FILE EDIT PROGRAM...
Enter Input Filename:  6165
Enter Output Filename:  6165A
```

MANEDIT then prompts the user for the commands defined in section 7.2 to examine or modify manuscript header parameters. A listing is generated in the file "H6165A.LIS".

VIII. Program REVIEW

8.1 Program Description

REVIEW is an interactive program used to examine the contents of the review file generated by CHECK or VERIFY when either of these programs is used in automated mode. The review file contains both the original and revised versions of all features with automatically corrected intersections. The contents can be expanded to include all automatically corrected exit point errors, if desired, by setting the parameter REVWSEV to the appropriate value as CHECK or VERIFY is executed. REVIEW works much like EDIT in that feature points can be examined, and features can be plotted fully, partially, or relative to the containing manuscript on the user's graphics terminal. The user can thus perform detailed comparisons of the original and corrected features in the vicinity where automated corrections occurred.

The use of REVIEW is imperative whenever automated corrections are performed. REVIEW is the only means by which the user can verify that automated corrections are indeed acceptable. If all review file features contain acceptable automated corrections, the user takes no further action for these features, since the same automated corrections exist on the intermediate DFAD tape (in the case of CHECK), or in the output edit file (in the case of VERIFY). If however, the user discovers an unacceptable automated correction in a review file from CHECK, he must take action to insure that the subject feature is included in the edit file for interactive correction. If an unacceptable correction is found in a review file generated by VERIFY, the user should delete the output edit file and correct the problem interactively.

REVIEW asks the user only for the review filename. The program then promptS the user for the interactive commands discussed below.

8.2 REVIEW Commands

The REVIEW program has 19 user commands which can be used to review automated corrections and to compare the original to the corrected feature. Many of the commands are similar to those used by EDIT. If parameters are not supplied with commands that require them as the user issues the comma d, REVIEW prompts for them. Each command is addressed in detail in Table VII.

TABLE VIII

REVIEW Commands

Command	Form/Description
*******	****************
AL	AVERAGE LENGTH TEST [AL (i,j)] This command performs the average length test on the current feature. Optional parameters i and j are the average length factor and average length distance, respectively. If i and j are not supplied, the default values of 4 and 10 are used.
CA	CHECK ALL [CA] This command allows the user to check the current feature for points outside the manuscript, exit point errors, average length errors, and intersections. If the feature is larger than 1000 points, REVIEW will give the user the chance to abort the intersection test before it starts.
CE	CHECK FOR EXIT POINT ERRORS [CE] This command tests the current feature for exit point errors.
CL	CHECK LOCAL [CL (i,j)] This command allows the user to check a portion of a feature (between points i and j) for intersections.
DD	DUMP DATA [DD] This command allows the user to dump data for the current feature to the listing file.
DM	DIAMOND MARKER [DM (i)] This command draws a diamond around the specified point on a plot of the current feature on the user's graphic terminal. Parameter i is the index of the point to be marked.
EP	EXAMINE POINT [EP (i)] This command allows the user to examine the latitude/longitude deltas of the specified point and up to 10 surrounding points. Parameter i is the index of the point of interest.
EX	EXIT [EX] This command exits the program.

GN	GET NEXT FEATURE [GN] This command retrieves the next feature from the review file.
HE	HELP [HE] This command lists all review commands at the user terminal.
MO	MODE [MO (i)] This command explicitly establishes the default mode for the current feature. If i=1, the default mode is set to "original", meaning that all subsequent commands will operate on the original feature definition until the MODE command is issued again, or until the mode is set in conjunction with another command (i.e., PAC = Plot All Corrected, CLO = Check Local Original). If i=2, the default mode is set to "corrected".
PA	PLOT ALL [PA] This command plots all points of the current feature at the user's graphics terminal.
PL	PLOT LOCAL [PL (i,j,k,l)] This command plots lines from two non-contiguous segments of the current feature at the user's graphics terminal. Up to 1000 points may be plotted per segment. Parameters: i - first point of first segment j - last point of first segment k - first point of second segment l - last point of second segment
PM	PLOT MANUSCRIPT RELATIVE [PM] This command plots all points of the current feature relative to the containing manuscript at the user's graphics terminal.
PO	POINT OUTSIDE MANUSCRIPT CHECK [PO] This command checks the current feature for points which lie outside the manuscript boundary.
PP	PLOT PARTIAL [PP (i,j)] This command plots a segment of the current feature up to 2000 points in length at the user's graphics terminal.

SA	SHOW ARRAYS [SA] This command displays the number of points in the original and corrected feature arrays.
SD	SHOW DATA [SD] This command displays all header information for the current feature, along with the original number of points, original feature area, error severity, number of original intersections, exit point errors, and other errors, and manuscript boundary deltas.
SF	SKIP FEATURE [SF (m,f)] This command allows the user to skip to a desired feature in the review file by specifying the manuscript (m) and FAC (f) numbers.

8.3 Program Operating Instructions

The REVIEW program is initiated as follows:

```
$ REVIEW
DFADEDIT REVIEW PROGRAM...
Enter Review Filename: 6165
       .
       .
 (File Header Information Listed Here)
       .
       .
Command?
```

The "Command?" prompt signifies that REVIEW is ready to accept commands from the list defined in Table VIII to allow the user to sequence through the review file and examine all automated corrections.

IX. Program DIRECTRY

9.1 Program Description

DIRECTRY allows the user to create new directory files or modify existing files. As discussed in section 2.4.4, the primary use of the directory file is as a guide to FETCH to control the generation of the edit file. The user can change directory file entries to override unacceptable automated corrections. The user can create new directory file records to initiate correction of a feature not already included in the directory file. Finally, the user may wish to create his own directory file which will allow him to generate an edit file containing features of his own choosing. DIRECTRY provides all of these capabilities.

The use of DIRECTRY may not always be required in the error correction process for a given tape, especially if REVIEW determines that all automated corrections are acceptable. However, even in this case, the user may wish to use DIRECTRY to obtain a listing of the contents of the directory file. This listing serves as useful documentation of the errors or warnings associated with a particular tape.

It should be noted that DIRECTRY does not create a new version of the directory file each time it is executed. DIRECTRY modifies the input file in most cases, and only creates a new file when specifically told to do so by the user.

9.2 DIRECTRY Commands

DIRECTRY has eight commands which allow the user to operate on the contents of a directory file. Once the user has initiated the program and supplied the directory filename, DIRECTRY requests commands using the "Command?" prompt. Each command is discussed below. Parameters associated with some commands are shown in parentheses, and the program prompts for them when required if they are not provided with the command code. Note that directory file records are accessed by their associated manuscript and FAC numbers.

TABLE IX

DIRECTRY Commands

Command *******	Form/Description ****************
CS	CHANGE SELECTION CODE [CS (Man,FAC)] This command allows the user to change the selection code associated with the particular directory file entry for manuscript "Man" and FAC number "FAC". The selection code provided by CHECK for each feature entry is a null value (blank). For an entry created with the DIRECTRY command Insert Record (IR), the default selection code is "+", which means the feature should be included in the edit file regardless of the associated feature error severity and error severity threshold. The only other option for the selection code is "-", which signifies to FETCH that the feature should not be included in the edit file, regardless of error severity. The CS command allows the user to change the selection code for any directory file entry to any of these three values.
DR	DELETE RECORD [DR (Man,FAC)] This command allows the user to delete the entry for the specified manuscript and FAC from the directory file. (The same effect may be achieved for the purposes of generating the edit file by changing the selection code to "-".)
EX	EXIT [EX] This command exits the DIRECTRY program.
HE	HELP [HE] This command lists the DIRECTRY commands along with a brief definition at the user's terminal.
IR	INSERT RECORD [IR (Man,FAC)] This command allows the user to insert (create) a new directory file entry for the specified manuscript and FAC. The selection code is set to "+" (include regardless of severity).

PD PRINT DIRECTORY [PD]
This command writes the full contents of the directory file to a listing file "Dfile.LIS", where "file" is the filename supplied by the user in response to the DIRECTRY program prompt "Enter Directory Filename ". This file may be printed later with the VAX/VMS command "PRI Dfile.LIS".

SR SHOW RECORD CONTENTS [SR (Man,FAC)]
This command displays the contents of the specified directory file entry at the user's terminal.

TD TYPE DIRECTORY FILE [TD (Man)]
This command lists the contents of the directory file at the user's terminal. If "Man" is not provided, the entire file is listed. If "Man" is provided, only the directory file entries corresponding to the specified manuscript are listed.

9.3 Program Operating Instructions

The following is an example of an interactive session using DIRECTRY. Once the program is initiated, the user enters the filename, and the program begins to prompt for commands. The example illustrates changing a selection code for an unacceptable automated correction, inserting a new entry, and generating the directory file listing:

```
$ DIRECTRY
DFADEDIT DIRECTRY PROGRAM...
Enter Filename:  6165
Are You Creating A New Directory File?  N
       .
       .
 (File Header Information is Listed)
       .
       .
Command?  CS 3,5
Enter User Selection Code:
+ = Include Regardless of Severity
- = Exclude Regardless of Severity
```

```
N = Null Selection Code
X = Cancel Change Operation:  +
New Selection Code:
3-Man 5-FAC Include Regardless of Severity

Command?  IR 4,195
New Record Created:
4-Man 195-FAC Include Regardless of Severity

Command?  PD

Command?  EX
$ PRI D6165.LIS
```

The directory file at this point contains the modifications, and the listing will reflect these modifications.

X. Program FETCH

10.1 Program Description

The primary function of FETCH is to automatically generate an edit file given the original DFAD tape, the directory file generated by CHECK, and a user error severity threshold. A secondary purpose of FETCH involves the ability to interactively plot or dump data for specified features on a DFAD tape. In either case FETCH runs interactively, although the level of interaction is much lower for edit file generation, also known as directory mode. Directory mode is the only way to generate an edit file with FETCH.

Thus FETCH is usually run in directory mode, using the directory file generated by CHECK (and possibly modified by DIRECTRY) as a guide to select features from the original DFAD tape to be included in the edit file. When the user specifies directory mode, FETCH prompts for the output edit filename, and for the first input directory filename. FETCH allows for the use of multiple directory files to generate a single edit file, which is edited and ultimately used in the merge process to create the corrected output tape. FETCH also prompts the user for an error severity threshold. This threshold controls which directory file features are selected for the edit file. If the directory file entry has an error severity greater than or equal to the threshold value, the feature is included in the edit file. Thus, for standard edit file generation where automated correction was used by CHECK, the threshold value should be 4. This prevents an inadvertent override of automated corrections for features classified as error severity 2 or 3, but includes all features with warnings or more serious errors so that the user can inspect them. For example, although such features have been successfully corrected automatically, a threshold value of 3 would cause all automatically corrected intersections to be included in the edit file. These edit file features would then eventually replace the automatically corrected intersections from the intermediate DFAD tape during the merge process. The user must therefore be careful in selecting a threshold value when generating edit files using FETCH.

Threshold selection may be overridden by the explicit inclusion "+" or exclusion "-" selection codes, which may be associated with any feature. These selection codes may be set explicitly by the user with the DIRECTRY program "Change Selection" (CS) command.

In directory mode, the output listing is written to the file "Fedit.LIS", where "edit" is the user supplied edit filename provided in response to the "Enter Edit Filename " prompt. The listing file contains directory file header information, error severity threshold information, and a list of all included edit file features along with their error severity and explicit selection codes, if applicable.

When used to plot or dump features interactively, FETCH prompts the user for the manuscript and FAC numbers of the desired features. A manuscript and FAC entry of 0,0 in response to the prompt for the next feature will terminate the program in this mode. If a listing is generated, it appears in a file named "FETCHLIST.LIS".

10.2 Program Operating Instructions

The example below illustrates the use of FETCH in directory mode. Two directory files are used to generate an edit file, and the listing file is then printed.

```
$ MOU/FOR MTA0:
$ FETCH
DFADEDIT FETCH PROGRAM...
Enter Tape Unit (A/B)?  A
Directory Mode?  Y
Enter Edit Filename:  6165
Enter Directory Filename:  6165A
Enter Severity Threshold:  2
Severity  Threshold  Other  Than  4  May  Violate  Automated
Corrections
Do You Wish To Reconsider?  Y
Enter Severity Threshold:  4
         .
         .
 (File Header Information is Listed)
         .
```

```
       .
 (Selected Features are Listed)
       .
       .
Any More Directory Files?  Y
Enter Directory Filename:  6165B
Enter Severity Threshold:  4
       .
       .
 (File Header Information is Listed)
       .
       .
 (Selected Features are Listed)
       .
       .
Any More Directory Files?  N
$ PRI F6165.LIS
```

XI. <u>Program EDIT</u>

11.1 Program Description

EDIT is an interactive program which allows the user to operate on an input edit file using a number of commands, and generates an output file containing the modified feature data. The program first prompts the user for an input edit filename. If the user enters a valid filename, the file is opened and the file header data is displayed at the user's terminal. If "*" is entered in response to the prompt for the input edit filename, EDIT assumes that there is no input edit file and continues by prompting for an output edit file. If the user desires to simply examine an edit file, no output edit file is generated. However, if the user wishes to perform error correction, a new edit file is created with the user-supplied output edit filename. At this point, EDIT is ready to accept one of the 40 user commands defined in section 11.2. EDIT has a number of commands that provide interactive analysis and error-checking capabilities, so that the user may assure himself that errors have been effectively corrected.

Manuscript boundary information is carried along with each feature in the edit file to facilitate manuscript-relative plots, exit point tests, and tests for points outside the manuscript boundary. If a FAC 1 feature is encountered in the edit file, it's presence is likely due to the fact that it defines a skewed manuscript boundary. Thus whenever a FAC 1 feature is output, its current manuscript boundary definition is retained and replaces the manuscript boundary definition for any subsequent edit file features from the same manuscript. In this way, manuscript plots in EDIT, and exit point and point outside manuscript tests in EDIT and VERIFY can make use of the corrected manuscript boundary values.

All changes made by the user during an edit session are documented in the edit log, which is a listing file generated by edit. This file is named "Eroot.LIS", where "root" is the user-supplied filename for the output edit file. A listing may also be generated to simply dump feature data when no output file is being generated. The listing file in this case is named "EDITLOG.LIS".

Once the edit session is complete and all the features in the edit file have been corrected, the edit file is ready for processing by the VERIFY program to assure that all original errors have been corrected and no new errors have been introduced.

11.2 EDIT Commands

The EDIT program has 40 user commands which can be employed in the interactive analysis and correction of feature errors. EDIT allows the user to enter the accompanying parameters for many of the EDIT commands on the same line as the command keyword. If parameters are not supplied to the commands which require them, EDIT will prompt for them. Each command is addressed in detail below in Table X.

TABLE X

EDIT Commands

Command	Form/Description
*******	****************
AF	ADD FEATURE [AF] This command allows the user to insert a feature from another edit file into the current position in the current edit file. This command is operational only in edit mode and only when no other feature is contained in the working array.
AL	AVERAGE LENGTH TEST [AL (i,j)] This command performs the average length test on the current feature. Optional parameters i and j are the average length factor and average length distance, respectively. If i and j are not supplied, the default values of 4 and 10 are used.
CA	CHECK ALL [CA] This command allows the user to check the current feature for points outside the manuscript, exit point errors, average length spikes, and intersections. If the feature is larger than 1000 points, EDIT gives the user the chance to abort the intersection test before it starts.
CC	COPY CAPTURE [CC] This command allows the user to copy a captured portion of a feature stored in the capture array (described in the CF command below) to the working array for editing.
CE	CHECK FOR EXIT POINT ERRORS [CE] This command tests the current feature for exit point errors.
CF	CAPTURE FEATURE [CF (i,j)] This command "captures" a part of the current feature from the working array and copies that part to a separate capture array for temporary storage until the user is ready to operate on it. In addition, EDIT closes the captured feature and deletes the captured part of the feature from the working array. Through a sequence of commands as shown in the example below, the original feature will eventually be output as two separate

features with the same header information. MERGE will properly adjust FAC numbers when writing the output tape. This command is used to separate a feature with coincident segments along the manuscript boundary into two error-free features. The parameters i and j are the first and last points of the captured feature.
EXAMPLE:
CF - Capture part of the feature
SW - Save main feature
CC - Copy capture array to working array
CA - Check/correct the captured feature
OF - Output the captured feature
RM - Copy main feature to working array
CA - Check/correct the main feature
OF - Output the main feature

CH CHANGE HEADER [CH]
This command allows the user to change various parameters in the header of the current feature. The user should take care to insure that all changed parameters are valid, since EDIT does not fully verify that new header values are correct.

CL CHECK LOCAL [CL (i,j)]
This command allows the user to check a portion of a feature (between points i and j) for intersections.

CP CHANGE POINT [CP (i x,y)]
This command allows the user to change the latitude and longitude values of any point in the current feature. Parameter i is the index of the point to be changed, x is the new longitude delta, and y is the new latitude delta.

CX CHANGE X VALUE [CX (i,x)]
This command allows the user to change the longitude delta of a specified point in the current feature. Parameter i is the index of the point to be changed, and x is the new longitude delta.

CY CHANGE Y VALUE [CY (i,y)]
This command allows the user to change the latitude delta of a specified point in the current feature. Parameter i is the index of the point to be changed, and y is the new latitude delta.

DC DISREGARD CURRENT FEATURE [DC]
This command allows the user to flag the current feature to be disregarded in the MERGE process. An edit file feature thus flagged will not replace the corresponding feature from the input tape on MERGE output tape. If the feature is already flagged to be disregarded, the use of this command revokes the flag and returns the feature to its original status.

DD DUMP DATA [DD]
This command allows the user to dump data for the current feature to the listing file.

DF DELETE FEATURE [DF]
This command allows the user to flag the current edit file feature for deletion during the MERGE process. A feature thus flagged will not exist on the MERGE output tape. If there is no current feature when the user issues this command, EDIT prompts for a manuscript and FAC number, and creates a dummy edit file feature to trigger deletion of the requested feature in the MERGE process. If the user issues this command with a current feature already flagged for deletion, EDIT revokes the deletion flag and returns the feature to its original status.

DM DIAMOND MARKER [DM (i)]
This command draws a diamond around the specified point on a plot of the current feature on the user's graphics terminal. Parameter i is the index of the point to be marked.

DP DELETE POINT [DP (i)]
This command allows the user to delete one or more points of the current feature. Parameter i is the index of the point to be deleted. If i is not supplied, EDIT prompts the user for the index i along with the number of points to delete starting at i. In this way, the user can delete multiple points with one command.

EP EXAMINE POINT [EP (i)]
This command allows the user to examine the latitude/longitude deltas of the specified point and up to 10 surrounding points. Parameter i is the index of the point of interest.

EX	EXIT [EX] This command exits the program. If in edit mode, any features remaining in the input edit file are copied to the output edit file.
GN	GET NEXT FEATURE [GN] This command retrieves the next feature from the edit file. This command is not valid if EDIT is generating an output edit file and the current feature has not yet been output.
HE	HELP [HE] This command lists all edit commands at the user terminal.
IF	INSERT FEATURE [IF] This command allows the user to insert (create) a new feature at the current position in the edit file. EDIT prompts for all feature header parameters, and the user inserts the feature points using the IP (insert point) command.
IP	INSERT POINT [IP (i,x,y)] This command allows the user to insert a new latitude/longitude delta set in the position indicated by i. Parameters x and y are the new longitude and latitude deltas, respectively.
MP	MOVE POINTS [MP] This command allows the user to move points from one location to another within a feature.
NP	NEGATE POINT [NP (i)] This command sets or resets the exit point flag associated with point i of the current feature. The exit point flag is shown as set when the latitude value shown in the EP (examine point) command is negative.
OF	OUTPUT CURRENT FEATURE [OF] This c mmand is functional only when generating an output edit file, and causes the current feature (from the working arrray) to be output to the edit file.
PA	PLOT ALL [PA] This command plots all points of the current feature at the user's graphics terminal.

PC PLOT CAPTURE ARRAY [PC]
This command plots the contents of the capture array (the captured feature) at the user's graphics terminal.

PL PLOT LOCAL [PL (i,j,k,l)]
This command plots lines from two non-contiguous segments of the current feature at the user's graphics terminal. Up to 1000 points may be plotted per segment. Parameters:

- i - first point of first segment
- j - last point of first segment
- k - first point of second segment
- l - last point of second segment

PM PLOT MANUSCRIPT RELATIVE [PM]
This command plots all points of the current feature relative to the containing manuscript at the user's graphics terminal.

PO POINT OUTSIDE MANUSCRIPT CHECK [PO]
This command checks the current feature for points which lie outside the manuscript boundary.

PP PLOT PARTIAL [PP (i,j)]
This command plots a segment of the current feature up to 2000 points in length at the user's graphics terminal.

RD REVERSE DIGITIZATION [RD]
This command reverses the order of digitization for the current feature.

RM RETURN TO MASTER [RM]
This command copies the contents of the master array to the working array. It may be used to restore the working array after an unacceptable edit operation, or it may be used under normal circumstances in the capture process.

SA SHOW ARRAYS [SA]
This command displays the number of points in the master, working, and capture arrays.

SD SHOW DATA [SD]
This command displays all header information for the current feature, along with the original number of points, original feature area, error severity, number of original intersections, exit point errors, and other errors, and manuscript boundary deltas.

SF	SKIP FEATURE [SF (m,f)] This command allows the user to skip to a desired feature in the edit file by specifying the manuscript (m) and FAC (f) number. If an edit file is being generated, all skipped features are properly copied to the output edit file.
SW	SAVE WORKING ARRAY [SA] This command copies the working array to the master array. It is used during the capture process (see CF command) or as a method to save a partially edited feature.
SX	SEARCH X ARRAY [SX (x)] This command searches the longitude delta array of the current feature for occurrences of the user-specified longitude value x.
SY	SEARCH Y ARRAY [SY (y)] This command searches the latitude delta array of the current feature for occurrences of the user-specified latitude value y.
UF	UNWIND FEATURE [UF (i,j)] This command allows the user to "unwind" a general intersection. This allows resolution of some types of intersections while avoiding the deletion of feature points. The parameters i and j are the indices of the indicated intersection provided by an intersection detection algorithm in EDIT, CHECK, or VERIFY.

11.3 Program Operating Instructions

EDIT is an interactive program, initiated by the following command sequence:

```
$ EDITOR
DFADEDIT EDIT PROGRAM...
Enter Input Filename:  6165
       .
       .
 (File Header Information Listed Here)
       .
       .
Will An Output File Be Generated?  Y
Enter Output Filename:  6165A
```

Command?

EDIT is now ready to begin accepting commands interactively from the list provided in Table X. The particular commands used and the sequence in which they are issued depends upon the nature of the errors being corrected.

XII. Program VERIFY

12.1 Program Description

VERIFY is used to check and/or correct features contained in a DFADEDIT edit file. VERIFY is usually used to process an edit file which has been interactively corrected by the user with the EDIT program. In this mode, VERIFY is simply checking the features in the edit file for any errors that were bypassed or introduced during interactive error correction. VERIFY has the capability to perform automated error correction, and generates a corrected edit file when automated correction is specified. A review file is also created when automated correction is performed by VERIFY. This review file must be examined by REVIEW like the review file output by CHECK. Whether or not VERIFY operates in automated mode, all errors detected/corrected are documented in the listing file. As VERIFY processes selected features from the input edit file, it classifies each feature with an error severity code. These error severity codes have the same meaning as those used by CHECK (defined in Table VI). For more discussion on error severity codes, refer to section 5.1.2.1.

Processing continues with the same tests that are performed by CHECK on individual features. These include tests for stack and spike points, open areal and closed linear features, average length spikes, points outside manuscript, intersections, exit point errors, and other tests associated with skewed manuscripts and background features. Essentially all tests performed by CHECK, with the exception of manuscript header checks, are also performed by VERIFY. Automated corrections performed by VERIFY work in the same manner as those performed by CHECK. For details on any of these tests, refer to the appropriate discussion in Chapter 5.

Once the desired features in the input edit file have been processed, VERIFY closes the output edit and review files, if appropriate, and terminates normally. A summary for the run similar to that provided for CHECK is written to the listing file.

12.2 Operating Environment

VERIFY is a batch program and no interactive capability is provided. VERIFY is executed via a batch command file which is described in section 12.5.

The only peripheral requirements are access to the input edit file and a listing file, and sufficient space for the output edit and review files if automated correction is specified.

Processing time requirements for VERIFY are similar on a feature by feature basis to those of CHECK. That is, a large feature being tested for intersections takes about the same time to process in VERIFY as in CHECK. However, VERIFY is processing only the features in the edit file, and thus can verify that all interactive corrections have been implemented properly in much less time than would be required to reprocess the entire merged, corrected output tape with CHECK. Like CHECK, VERIFY has the capability to disable intersection detection/correction, so that corrections other than those involving intersections can be verified rapidly.

12.3 User Input and Control Parameters

The user provides a number of inputs to the VERIFY program, the first of which is the user filename for the input edit file. If automated correction is desired, the user also provides a user filename for the output edit file. This output user filename is also used for the output review file. Finally the user supplies a number of user control parameters to select various error correction options. These parameters are defined in Table XI below. The parameters are specified to VERIFY in the form "PARAMETER=value". For examples on entering these parameters, refer to section 12.5.

TABLE XI

VERIFY User Control Parameters

Parameter	Description
*********	***********
AFAREVAR	AREAL FEATURE AREA VARIATION This parameter is used in conjunction with automated intersection correction, and is a percentage of variation (increase or decrease) allowed during automated intersection correction for areal features. After each automated correction, the new feature area is computed, and if the area falls outside tha allowable limits, a fatal error (severity 5) is issued. The default is 10 percent, and permissible values range from 0 to 50.
AFPNTVAR	AREAL FEATURE POINT VARIATION This parameter is used in conjunction with automated intersection correction, and is a percentage of variation allowed in the number of points for areal features undergoing automated correction. If the number of feature points falls below the limit, a fatal error (severity 5) is issued. The default is 10, and the allowable range is 0 to 50.
AUTOMAX	MAXIMUM AUTOMATED CORRECTIONS (PER FEATURE) This parameter specifies the maximum number of automated corrections (exit points and intersections) allowed per feature. If automated correction attempts to exceed this number, the feature is assigned error severity 5, and automated correction is disabled for the feature. The maximum allowable value is the default value of 200.
AVGLDIST	AVERAGE LENGTH DISTANCE This parameter, along with AVGLFCTR, controls the average length test used to detect spike points. The default value is 10, and AVGLDIST is specified in tenths of arc-seconds. For details, refer to the discussion of the average length test in section 5.1.2.2.

AVGLFCTR	AVERAGE LENGTH FACTOR This parameter, along with AVGLDIST, controls the average length test used to detect spikes. AVGLFCTR is an allowed multiple of the average segment length between any two adjacent feature points computed for each feature. The default value is 4. For details, refer to the discussion of the average length test in section 5.1.2.2.
DELETDST	DELETION DISTANCE LIMIT This parameter controls the deletion of points when such action is required by automated intersection correction. DELETDST is specified by the user in tenths of arc seconds and is the maximum allowable perpendicular distance from a point to a line that joins its two adjacent points. For more details, see the discussion in section 5.3. The default value is 25.
FIRSTFAC	FIRST FAC This parameter is used in conjunction with FIRSTMAN to specify the first edit file feature to be processed by VERIFY. The default value is 1.
FIRSTMAN	FIRST MANUSCRIPT This parameter selects the first manuscript to be processed from the input edit file. The default value is 1.
INTERSCT	INTERSECTION DETECTION/CORRECTION SWITCH This parameter enables intersection detection if true, and disables intersection detection if false. Options are T (true) and F (false). The default value is true. Both INTERSCT and AUTOMODE must be TRUE to perform full automated error correction.
LASTFAC	LAST FAC This parameter is used in conjunction with LASTMAN to specify the last edit file feature to be processed by VERIFY. The default value is 16383.

LASTMAN	LAST MANUSCRIPT This parameter selects the last manuscript to be processed from the input edit file. If the user supplied value is larger than the manuscript of the last feature in the edit file, processing will terminate normally after the last edit file feature has been verified. The default value is 9999.
LFPNTVAR	LINEAR FEATURE POINT VARIATION This parameter is used in conjunction with automated intersection correction, and is a percentage of variation allowed in the number of points for linear features undergoing automated correction. If the number of feature points falls below the limit, a fatal error (severity 5) is issued. The default is 10, and allowable values range from 0 to 50.
LFPNTWRN	LINEAR FEATURE POINT WARNING THRESHOLD This parameter specifies that VERIFY will issue a warning (error severity 4) for any linear feature with more than LFPNTWRN points. The default value is 200.
PFPNTWRN	POINT FEATURE POINT WARNING THRESHOLD This parameter specifies that VERIFY will issue a warning (error severity 4) for any point feature with more than PFPNTWRN points. The default value is 100.
REVWSEV	REVIEW FILE THRESHOLD (LOWER LIMIT) This parameter allows the user to control the contents of the review file generated by VERIFY when AUTOMODE is true. The default value is 3, which means that only features with error severity 3 (automatically corrected intersections) are placed in the review file. The user may set REVWSEV equal to 2, in which case features with automatically corrected exit point errors are also placed in the review file. These (2 and 3) are the only allowable values for REVWSEV.
UNWINDST	UNWIND DISTANCE LIMIT This parameter controls the "unwind" algorithm used in the automated correction of some types of intersections. It is specified in tenths of arc seconds, and has a default value of 25. For more discussion, see section 5.3.

12.4 Program Output

VERIFY produces a listing file for each run. The listing contains information on input and output filenames as applicable, user control parameters, error messages similar to those issued by CHECK, and a summary of errors and processing for the entire run.

If automated correction is selected in VERIFY, an output edit file and an output review file are also produced. The output edit file contains all features contained on the input edit file, even though not all features are necessarily verified or corrected. The review file, like the review file generated by CHECK, contains both the original and corrected versions of all features which are successfully corrected automatically by VERIFY.

12.5 Program Operating Instructions

Since VERIFY is exclusively a batch program, a command file must be used to run the program. The command file is created with a system text editor, and is submitted to the batch job stream with the VAX/VMS "SUBMIT" command. A sample command file is provided below. The example provides for processing the features in manuscripts 2 through 7, without automated correction and without intersection detection. As in previous examples, the text to the right of the "!" in each line is a comment provided or information only, and should not be included in the command file.

```
$ VERIFY              !Run VERIFY
6165A                 !Input edit filename
                      !No output file - no auto correction
FIRSTMAN = 2          !User control parameters
LASTMAN = 7
INTERSCT=F
END
$ PRI V6165A.LIS      !Print listing file
```

XIII. Program MERGE

13.1 Program Description

MERGE is usually the last step in the error correction process for a given tape. If any manuscript header parameters have been changed, or if any features required interactive editing, MERGE is used to insert the modified or new data in the proper position on the output tape. This output tape then replaces the original DFAD tape.

MERGE accomplishes feature insertion, deletion, or replacement by continuously comparing the manuscript and FAC numbers of the next feature in the corrected input edit file with the current position on the input tape. When MERGE locates the proper tape position, it interrogates a disposition flag set for the edit file feature by the FETCH or EDIT program. This disposition flag is transparent to the user, but it determines whether the edit file feature is a replacement feature, whether it is completely new, whether it should be ignored, or whether it is simply a dummy feature to trigger deletion of a feature which exists on the tape.

MERGE may encounter a fatal error as a result of feature sequence problems in the edit file. These errors occur when edit file features are out of sequential manuscript and FAC order, and thus cannot be properly merged. The only recourse when a MERGE sequence error occurs involves returning to a version of the edit file which was created prior to the introduction of the error, and repeating the interactive edit process from that point. Sequence errors are introduced during the EDIT process, usually as a result of user misuse of the add feature (AF) or insert feature (IF) commands.

MERGE merges the contents of one edit file and one manuscript header file with one DFAD source or intermediate tape to produce the corrected output tape. Both disk files are optional, and if neither is provided, MERGE in effect simply copies the input tape to the output tape.

MERGE may be run interactively or in batch mode. An example command sequence used to run MERGE in batch mode will be provided in section 13.4.

MERGE requires only a comment line, manuscript header filename (if desired), edit filename (if desired), and listing filename from the user. MERGE always assumes that the input tape is on drive MTA0: and the output tape is on drive MTB0 .

13.2 Program Output

MERGE produces only two outputs. These are the corrected DFAD tape and a listing which documents all replacement, insertion, or deletion actions which occur during the MERGE process. The listing contains a summary for each manuscript as well as one for the entire run. Each feature that has undergone any major editing or other change is listed along with its original and current FAC numbers, original and current number of points, and a number of other error counts and header parameters. The output tape is in standard DFAD format, and except for changes due to interactive or automated error correction, is an exact copy of the input tape.

13.3 Program Operating Instructions

MERGE can be run either interactively or in batch mode. Since batch mode is the preferred method, an example command file is provided below. The interactive version (MERGEI) requires identical inputs, and prompts for them at the beginning of the run. In the batch command file example below, once again the text following the "!" on any given line is a coment, and should not be included in the actual command file.

```
$ MOU/FOR MTA0:        !Input tape (always)
$ MOU/FOR MTB0:        !Output tape (always)
$ MERGE                !Batch version of MERGE
TAPE S-6165            !Comment
6165A                  !Edit filename
                       !No manuscript header file for MERGE
6165M                  !List filename
$ PRI M6165M.LIS       !Print listfile
$ DIS MTA0:            !Dismount input tape
$ DIS MTB0:            !Dismount output tape
```

A blank field in batch mode, or a carriage return in interactive mode, for either the edit filename or the manuscript header filename indicates that no such file is to be used in the MERGE process. As in a batch execution of CHECK, the tapes should be physically mounted and on-line on the appropriate drives prior to issuing the "SUBMIT" command to run the batch job. The tape drives must not be allocated or mounted by any user, including the user who issues the SUBMIT command.

APPENDIX B

DFAD Feature Identification Codes

APPENDIX B

DFAD Feature Identification Codes

This appendix defines the 264 feature identification codes which are used to identify features in the DMA DFAD. The codes are grouped into the general categories of industry, transportation, commercial/recreational, residential/agricultural, communications, governmental/institutional, military/civil installations, storage, and landforms/vegetation.

Feature Identification	Code No.
Industry	
Extraction Industry (General)	101
Quarry	102
Gas/Oil Derrick	103
Offshore Platform	104
Offshore Platform with Derrick	105
Mine Shaft Superstructure	106
Processing Industry (General)	110
Chemical Processing Plant	111
Metal Processing Plant	112
Sewage Treatment Plant	113
Evaporator Mining (Settling Basins, Sludge Ponds)	114
Coke Plant	115
Blast Furnace	116
Refinery	120
Catalytic Cracker	121
Flare Pipe	122
Power Plants (General)	130
Hydro-electric Power Plant	135
Thermal Power Plant	136
Transformer Yard	137
Substation	138
Solar Energy Electrical Collection Panels	145
Solar Energy Heat Collection Panels	146
Heavy Fabrication Industry (General)	150
With Flat Roof	151
With Flat Roof and Monitor	152
With Gable Roof (Pitched)	153
With Gable Roof (Pitched) and Monitor	154
With Sawtooth Roof	155
With Curved Roof	156
Light Fabrication Industry (General)	160
With Flat Roof	161
With Flat Roof and Monitor	162
With Gable Roof (Pitched)	163
With Gable Roof (Pitched) and Monitor	164
With Sawtooth Roof	165
With Curved Roof	166

Feature Identification	Code No.
Disposal (General)	170
Scrap Yard	171
Metal Ore Slag Dump	172
Tailings, Waste Piles	173
Strip Mine	174
Associated Structures (General)	180
Building	181
Smokestack	182
Conveyor	183
Pumping Station	184
Bridge Crane	185
Rotating Crane	186
Rotating Crane on Tower	187
Cooling Tower	188
Hopper	189
Dredge, Powershovel, Dragline	190
Transportation	
Railroads (General)	201
Elevated	202
Railroad Yards/Sidings (Moderate to Heavy Activity)	203
Railroad Yards/Sidings (Light Activity)	204
Railroad Yards/Sidings (Empty)	205
Storage and Repair Building	220
Railroad Terminal Building	221
Railroad Station	222
Railroad Control Towers	223
Roundhouse	224
Roads (General)	240
Elevated	244
Causeways	245
Bridges (General)	260
Suspension	261
Cantilever	262
Arch	263
Truss	264
Moveable Span	265
Bridge Towers	266
Deck	267

Feature Identification	Code No.
Conduits (General)	280
Pipelines (Above Ground)	281
Aqueducts	282
Penstocks, Flumes	283
Associated Structures (General)	290
Commercial/Recreational	
Commercial Buildings (General)	301
With Flat Roof	302
Circular with Flat Roof	303
With Gable Roof	304
With Curved Roof	305
Recreational Activities (General)	320
Enclosed Stadium	321
Open-ended Stadium	322
Dome Stadium	323
Grandstand	324
Athletic Field	325
Amusement Park	330
Roller Coaster	331
Ferris Wheel	332
Artificial Mountain	334
Display Signs (General)	340
Advertising Billboards	341
Scoreboard	343
Overhead Highway Sign	344
Associated Structures (General)	350
Drive-In Theater Screens	352
Residential/Agricultural	
Multi-Family Dwellings (General)	401
Apartments/Hotels with Flat Roof	402
Apartments/Hotels with Gable Roof	403
Single Family Dwellings (General)	420
Mobile Home	421

Feature Identification	Code No.
Agricultural Buildings (General)	430
Stockyard/Holding Pen	433
Windmill (Truss)	434
Windmill (Solid)	435
Glass Houses (Green Houses)	436
Associated Structures (General)	450
Cemetery Buildings	451
Communications/Transmission Facilities and Tower Types	
Communication Towers (General)	501
Radio/Television, Tower Type "A"	511
Radio/Television, Tower Type "I"	512
Microwave Tower, Type "A"	520
Microwave Tower, Type "I"	521
Miscellaneous Towers (General)	530
Observation Tower	531
Tower on Structure	532
Athletic Field Lights	535
Electrified Railroad Gantries/Pylons	536
Power Transmission Towers (General)	540
Powerline Pylons, Type "A"	541
Powerline Pylons, Type "H"	542
Powerline Pylons, Type "I"	543
Powerline Pylons, Type "Y"	544
Associated Structures	560
Communication Buildings	561
Governmental and Institutional	
Governmental (General)	601
Capital Building	602
Administration Building	603
Prison	604
Palace	605
Castle	606
Institutional (General)	610
School	620
With Flat Roof	621
With Gable Roof	622

Feature Identification	Code No.
Hospital	630
With Flat Roof	631
With Gable Roof	632
Observatory	640
With Dome Roof	641
Houses of Religious Worship (General)	650
Associated Structures (General)	680
Steeple	681
Monument/Obelisk	682
Arch	683
Pyramid	684
Military/Civil Installations	
Airport/Airbase (General)	701
Control Tower	702
Terminal/Base Operations	703
Hangar with Flat Roof	704
Hangar with Curved Roof	705
Runways and Taxiways	706
Aircraft Parking Areas	707
Airport/Airbase Electronic Navigation Aids (General)	710
Radar Reflectors, Uni-Directional	711
Radar Reflectors, Bi-Directional	712
Radar Reflectors, Omni-Directional	713
VOR/VORTAC/TACAN Facility	714
Radar Antenna with Radome	715
Radar Antenna, Tower Mounted with Radome	716
Radar Antenna	717
Radar Antenna, Tower Mounted	718
Radar Antenna, Vehicle Mounted	719
Approach Lights Framework	720
GCA Facility	721
Runway Approach Lighting Systems (General)	725
Maritime Features (General)	750
Breakwater/Jetty	751
Wharf/Pier	752
Drydock	753
Lock	754
Off-Shore Loading Facilities	755
Exposed Wreck	756

Feature Identification	Code No.
Maritime Navigation Aids (General)	760
Navigation Light-Ship (Permanently Moored)	761
Lighthouse	765
Ground (General)	770
Barracks, Curved Roof	771
Barracks, Flat Roof	772
Barracks, Gable Roof	773
Motor Pools	774
Garage, Curved Roof	775
Garage, Flat Roof	776
Garage, Gable Roof	777
Depots	778
Associated Structures (General)	790
Administration Building	791
Engine Test Cells	797
Wind Tunnels	798
Storage	
Tanks (General)	801
Cylindrical, Flat Top	802
Cylindrical, Dome Top	803
Cylindrical, Peaked/Conical Top	804
Cylindrical, Peaked/Conical Top, Tower Mounted	805
Spherical	806
Spherical with Column Support	807
Blimp	810
Bullet	811
Telescoping Gasholder (Gasometer)	812
Closed Storage (General)	820
Silo	821
Grain Elevator	822
Grain Bin	823
Water Tower (Building)	824
Ordnance Storage Mounds	825
Open Storage (General)	830
Mineral	831

Feature Identification	Code No.
Associated Features (General)	860
Warehouses	861
Vehicle Storage Area	862
Vehicle Parking Area	863
Aircraft Storage Area	864
Ship Storage Area	865
Landforms, Vegetation and Miscellaneous Features	
Ground Surface (General)	901
Soil	902
Sand/Desert	906
Sand Dunes	907
Marsh/Swamp	908
Rice Paddies	909
Smooth Solid Rock	910
Boulder Field/Lava	911
Rocky Rough Surface	912
Dry Lake	913
Mud/Tidal Flats	914
Islands	915
Surface Features (General)	920
Levees/Embankments	921
Walls	922
Cuttings	923
Cliffs	924
Dams/Weirs	925
Water Intake Towers	926
Fences	927
Plaza/City Squares	928
Salt Water (General)	930
Salt Water, Sea State	931
Salt Water, Sea State, Subject to Ice	932
Salt Water, Subject to Ice	933
Salt Pans	934
Fresh Water (General)	940
Fresh Water, Sea State	941
Fresh Water, Sea State, Subject to Ice	942
Fresh Water, Subject to Ice	943
Waterfall	944

Feature Identification	Code No.
Vegetation (General)	950
Orchards/Hedgerows	951
Deciduous Trees	952
Evergreen Trees (including Mangrove and Nipa)	953
Mixed Trees (Deciduous and Evergreen)	954
Tundra	955
Vineyards	956
Snow/Ice Areas (General)	960
Permanent Snow	961
Permanent Ice -- Glacier/Ice Cap	962
Pack Ice (Temporary)	963
Polar Ice Pack (Permanent) August	964
Polar Pack Ice (Seasonal Limit) November	965
Polar Pack Ice (Seasonal Limit) February	966
Polar Pack Ice (Seasonal Limit) April	967
Regional Features (General)	980

APPENDIX C

Areal Feature Area Calculation

APPENDIX C

Areal Feature Area Calculation

This appendix addresses the area calculation for areal features and the conversion of the calculated area to a useful value in square nautical miles. The material presented here is an overview, and the reader should refer to Pitman (Ref 4: App B) for more detail.

As stated in the text, the area A of a polygon with n points, represented by $P_1(x_1,y_1), P_2(x_2,y_2), \ldots P_n(x_n,y_n)$, may be calculated by:

$$A = \frac{1}{2}[x_1y_2+x_2y_3+\ldots+x_{n-1}y_n+x_ny_1-y_1x_2-y_2x_3-\ldots y_{n-1}x_n-y_nx_1] \qquad \text{(C-1)}$$

Since DFAD areal features are polygons, the calculation of their area by this formula is straightforward. However, since the feature delta sets are defined in tenths of seconds of arc, the resulting area is in square tenths of arc-seconds. This is not a useful form, and it is thus necessary to convert to square nautical miles. An additional complication arises from the oblate spheroid shape of the earth, which ultimately leads to the use of separate conversion factors for distances measured along a meridian of longitude verses distances measured in a direction perpendicular to the meridian.

The approach taken here is to approximate conversion factors for both latitude and longitude as a function of the DMA DFAD latitude defined on the World Geodetic System (WGS). These conversion factors are calculated once per manuscript using an average manuscript latitude computed from the north and south boundaries of the current manuscript. Figure C-1 distinguishes the geocentric latitude (θ) from the geodetic

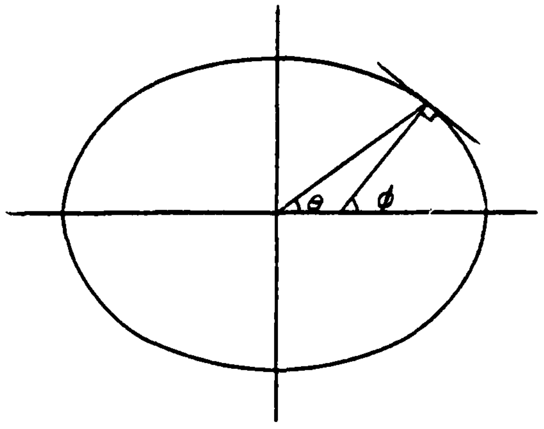


Fig C-1. Geocentric/Geodetic Latitude

latitude (θ). Geocentric latitude is the latitude based on the angle from the earth's center, while geodetic latitude is related to the perpendicular to the tangent of the earth's local surface. From Pitman (Ref 4: App B), the angle between the geodetic latitude and the geocentric latitude is given by:

$$\tan(\phi-\theta) = -\frac{1}{R}\frac{dR}{d\theta} \qquad \text{(C-2)}$$

$$= \frac{e \sin 2\theta}{1-e \sin^2\theta} \qquad \text{(C-3)}$$

where

R = the radius of the earth

e = the ellipticity of the earth

Ellipticity is calculated as:

$$e = 1 - \frac{b}{R_o} \qquad \text{(C-4)}$$

where

b = the polar radius of the earth

R_o = the equatorial radius of the earth

Ellipticity for the WGS spheroid used by the DFAD is defined as 1/298.26. Eq (C-3) can be approximated as:

$$\phi - \theta = e \sin 2\theta \qquad \text{(C-5)}$$

Through some manipulation and series expansion, and remembering that e is small with respect to 1, Eq (C-5) may be rewritten as:

$$\theta \simeq \phi - e \sin 2\phi \qquad \text{(C-6)}$$

Eq (C-6) is thus an adequate approximation for geocentric latitude as a function of geodetic latitude.

Pitman further defines the radius of curvature along a meridian as:

$$\rho_{LAT} = R_o[1-e(2-3 \sin^2 \theta)] \qquad \text{(C-7)}$$

where R_o is the equatorial radius of the earth. For the WGS spheroid, R_o is approximately 20925640 feet.

Pitman develops a relationship which defines the radius of the earth as:

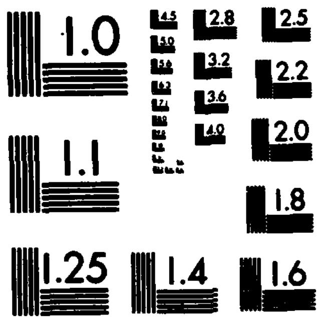
$$R = R_o(1-e \sin^2 \theta) \qquad \text{(C-8)}$$

The local radius of curvature in a direction perpendicular to the meridian can thus be approximated as:

$$\rho_{LON} = R_o(1-e \sin^2 \theta)\cos \theta \qquad \text{(C-9)}$$

AD-A124 893   IMPROVING THE USEABILITY OF THE DEFENSE MAPPING AGENCY   3/3
DIGITAL FEATURE AN..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI..   M R NICOL
UNCLASSIFIED   SEP 82 AFIT/GCS/EE/82S-9   F/G 9/2   NL

END
DATE
FILMED
83
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

where the cos(θ) term provides the shortening factor which accounts for decreasing meridian spacing with increasing latitude. This radius of curvature is approximated by a circle created by passing a plane through the geoid, parallel to the equator, at the local latitude. It is not to be confused with (ρ_λ) as defined by Pitman.

Thus the area of an areal feature is calculated as:

$$\text{Area} = \frac{A/(\rho_{LAT} \cdot \rho_{LON})}{6076^2} \qquad \text{(C-10)}$$

where A is the polygonal area from Eq (C-1). The denominator in Eq (C-10) converts from square feet to square nautical miles.

Although several approximations are employed in this area calculation technique, the result is adequate for application in the DFADEDIT software. The error correction algorithm monitors changes in feature area as it attempts automated correction, and the absolute accuracy of the area is not critical.

APPENDIX D

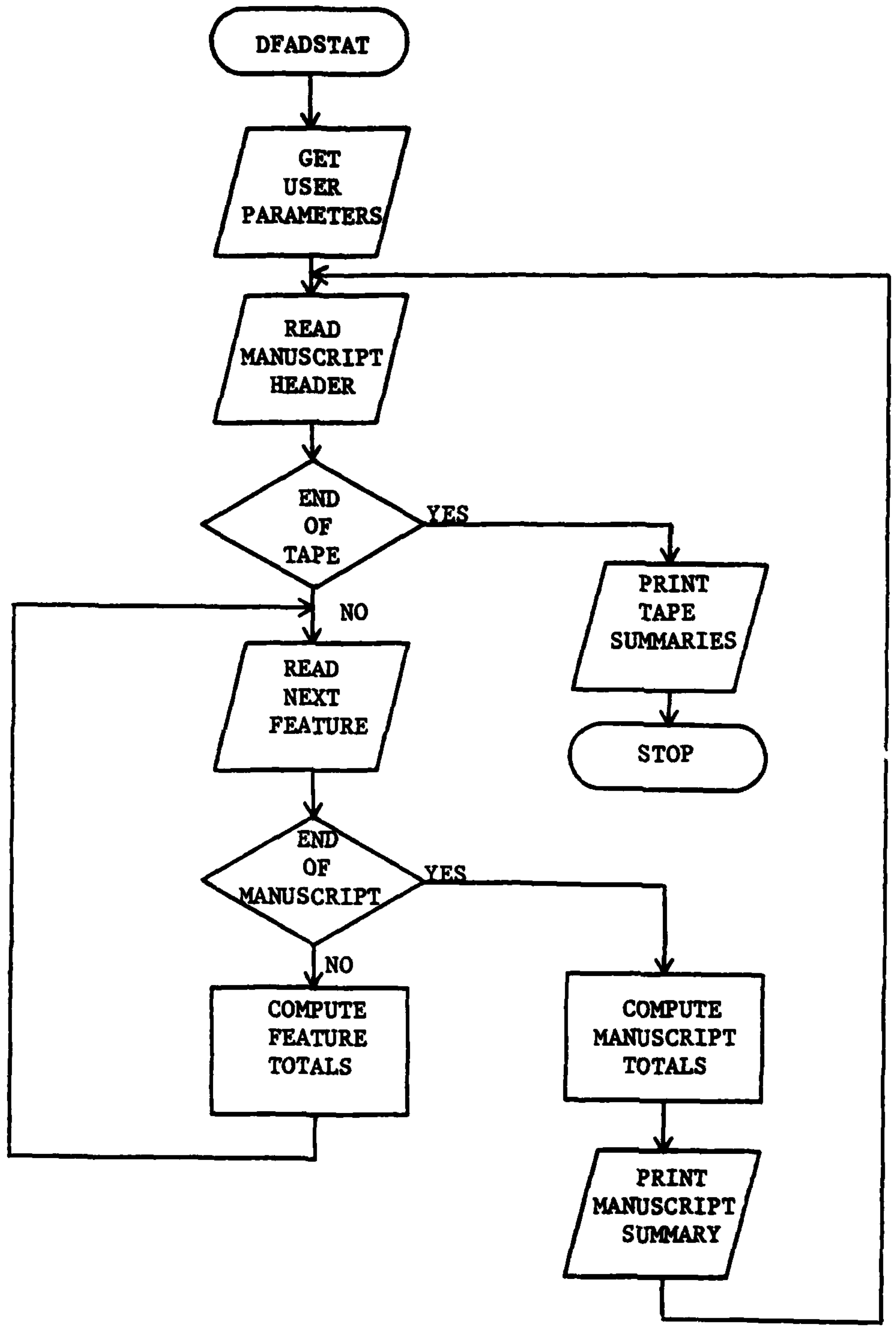
System Flow Charts

APPENDIX D

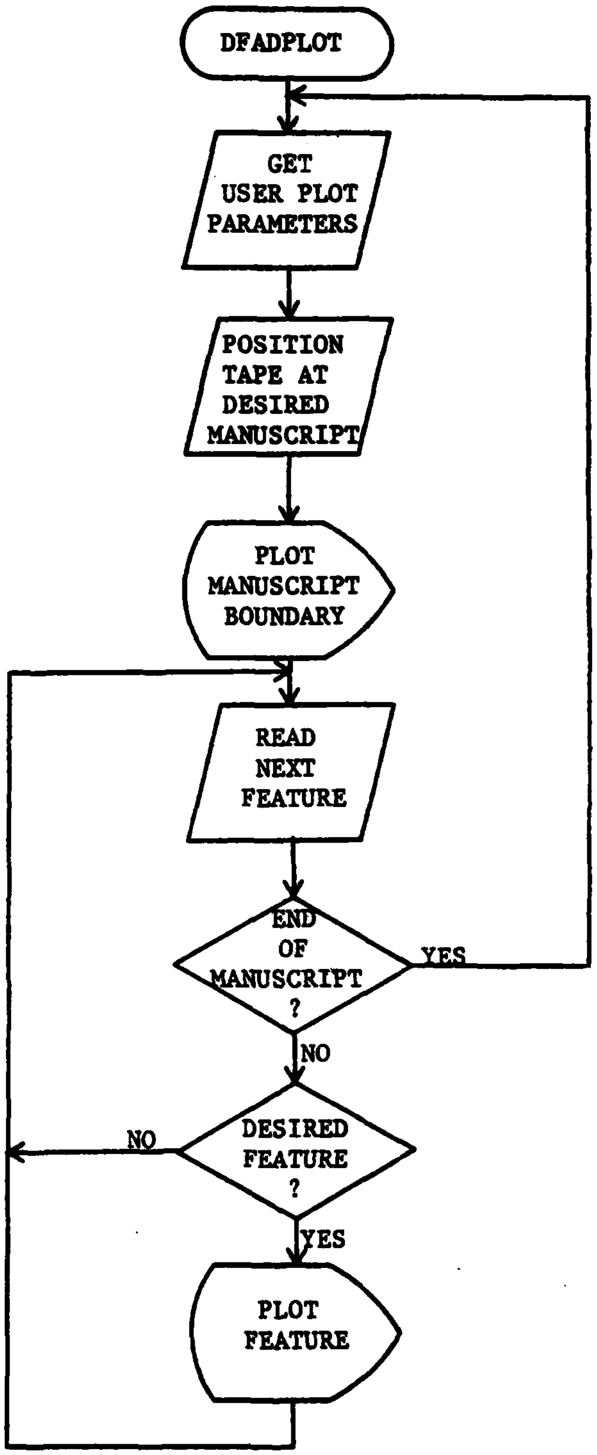
System Flow Charts

The DFADEDIT software consists of eleven main programs and numerous subroutines, some of which are common to two or more of the eleven main programs. This appendix contains a flowchart for each of the main programs. These are top-level flowcharts, and thus only the general functions and primary lines of communication are shown.

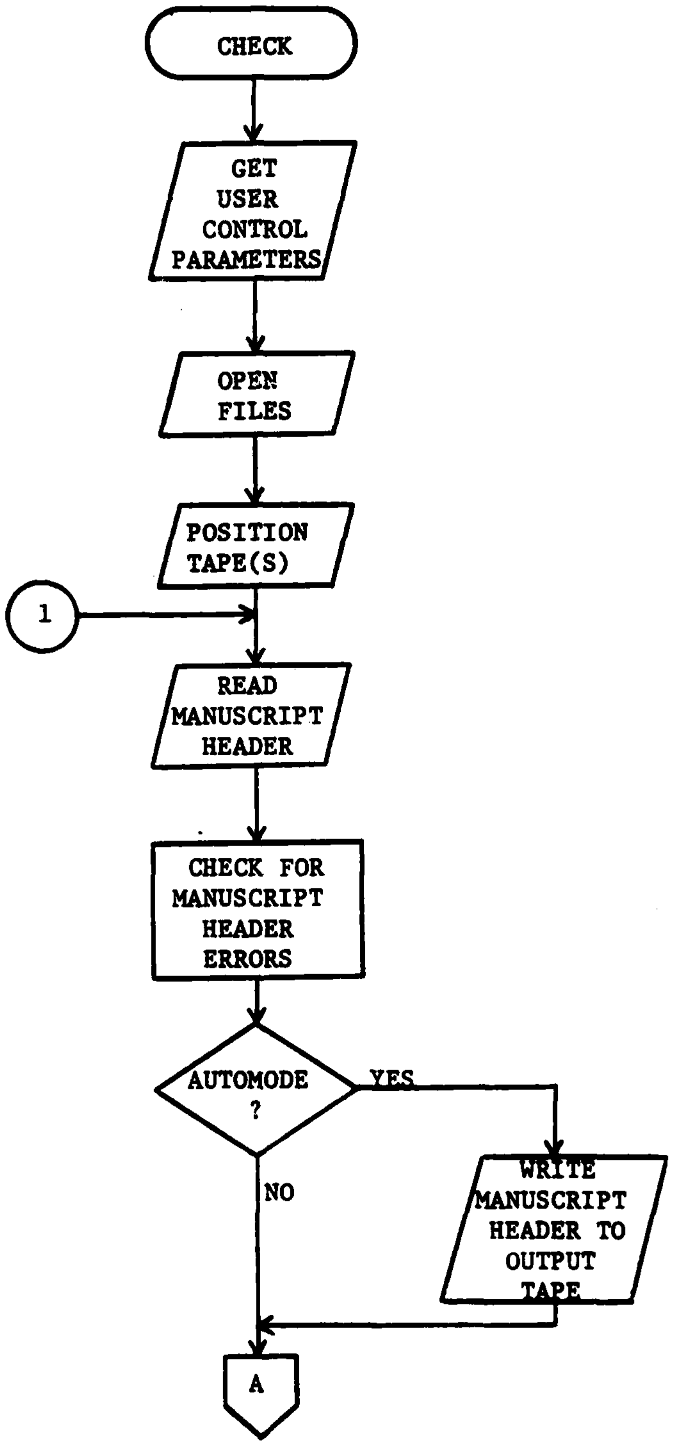
PROGRAM DFADSTAT



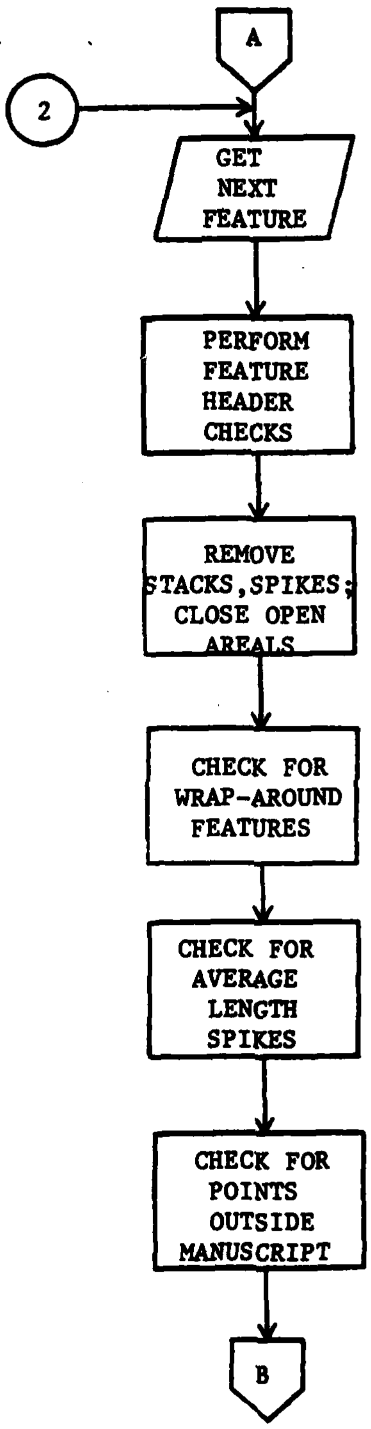
PROGRAM DFADPLOT



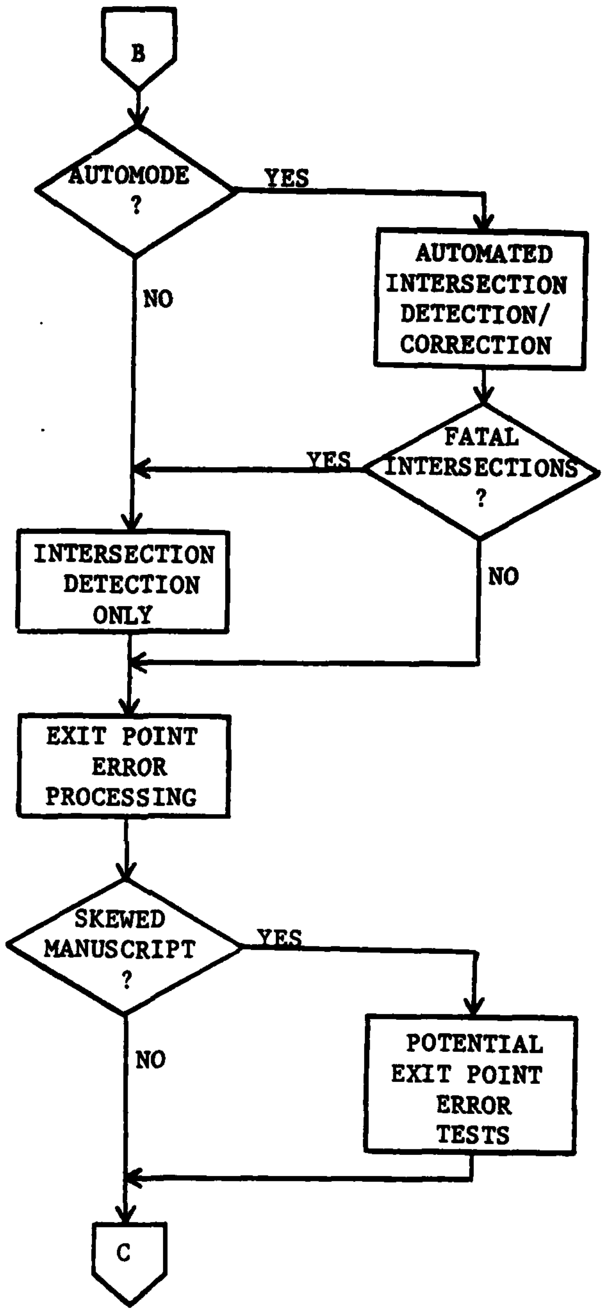
PROGRAM CHECK

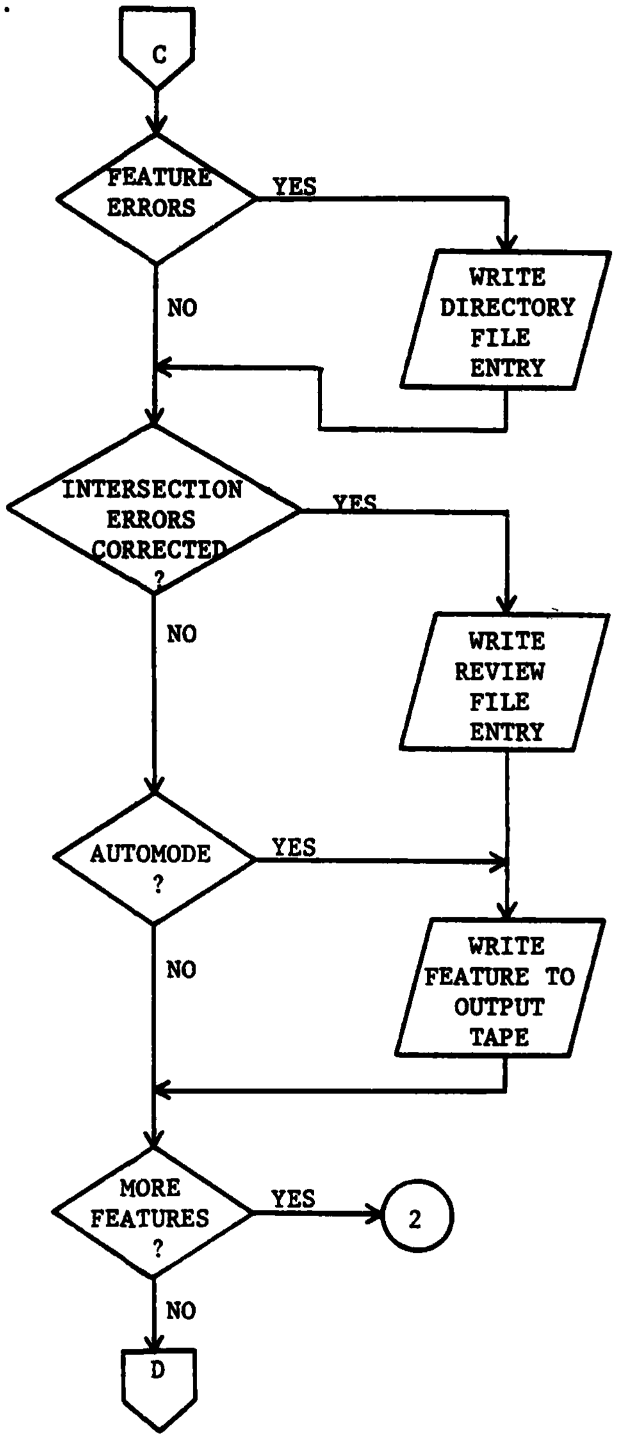

CHECK
GET USER CONTROL PARAMETERS
OPEN FILES
POSITION TAPE(S)
1
READ MANUSCRIPT HEADER
CHECK FOR MANUSCRIPT HEADER ERRORS
AUTOMODE ?
YES
NO
WRITE MANUSCRIPT HEADER TO OUTPUT TAPE
A

PROGRAM CHECK (cont'd)

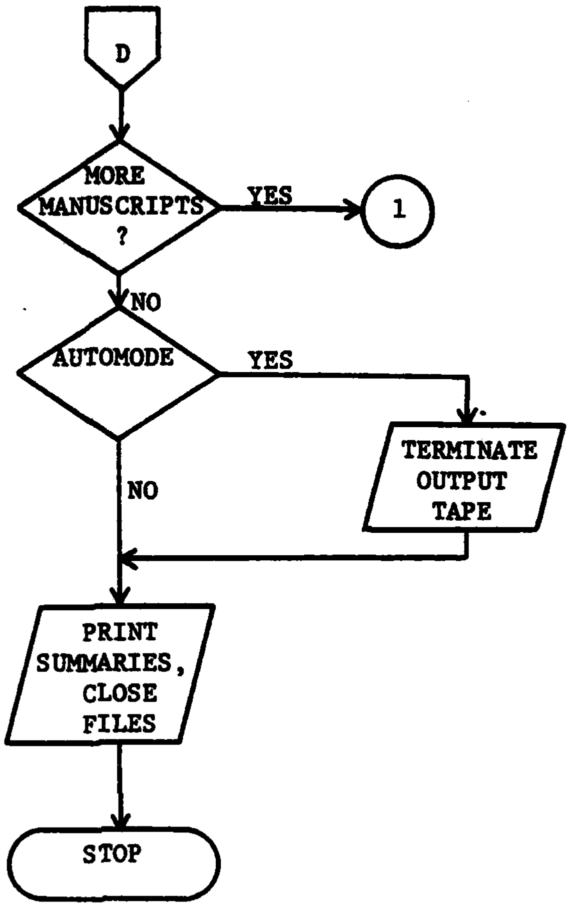

A
2
GET NEXT FEATURE
PERFORM FEATURE HEADER CHECKS
REMOVE STACKS, SPIKES; CLOSE OPEN AREALS
CHECK FOR WRAP-AROUND FEATURES
CHECK FOR AVERAGE LENGTH SPIKES
CHECK FOR POINTS OUTSIDE MANUSCRIPT
B

PROGRAM CHECK (cont'd)

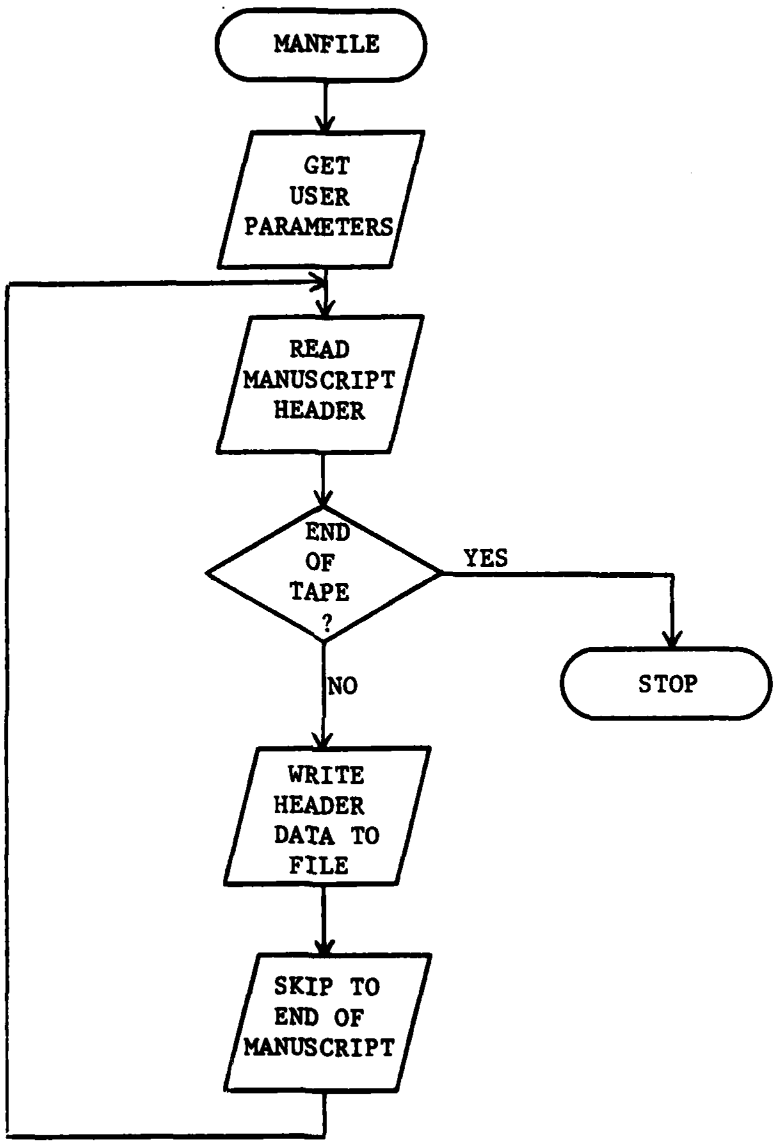

B
AUTOMODE ?
YES
NO
AUTOMATED INTERSECTION DETECTION/ CORRECTION
FATAL INTERSECTIONS ?
YES
NO
INTERSECTION DETECTION ONLY
EXIT POINT ERROR PROCESSING
SKEWED MANUSCRIPT ?
YES
NO
POTENTIAL EXIT POINT ERROR TESTS
C


C
FEATURE ERRORS
YES
WRITE DIRECTORY FILE ENTRY
NO
INTERSECTION ERRORS CORRECTED ?
YES
WRITE REVIEW FILE ENTRY
NO
AUTOMODE ?
YES
WRITE FEATURE TO OUTPUT TAPE
NO
MORE FEATURES ?
YES
2
NO
D

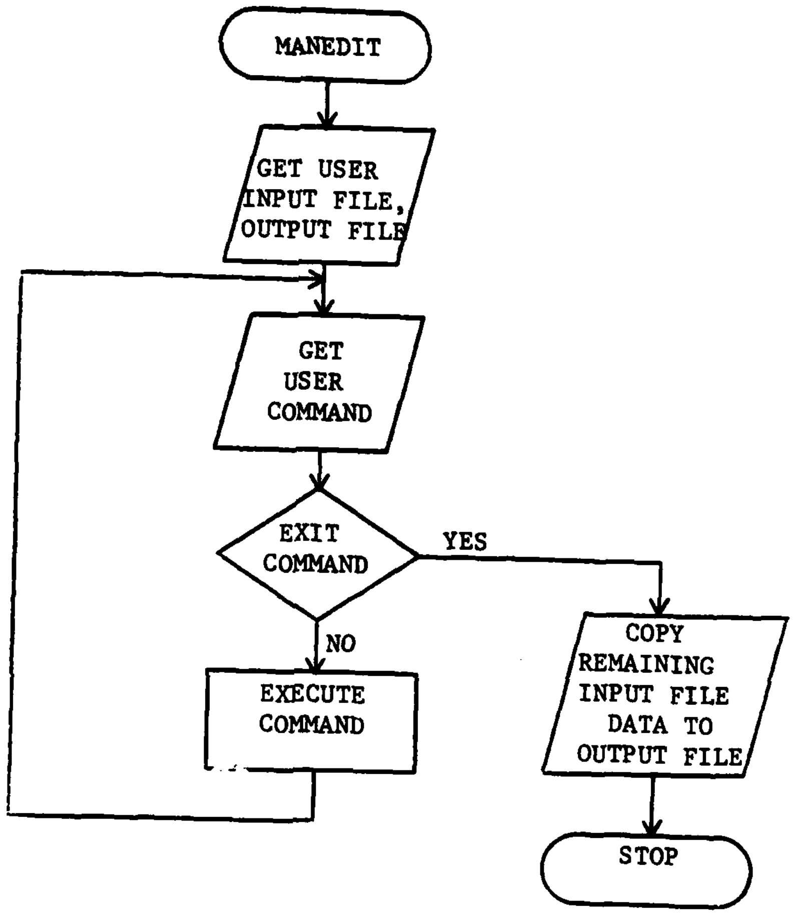
PROGRAM CHECK (cont'd)



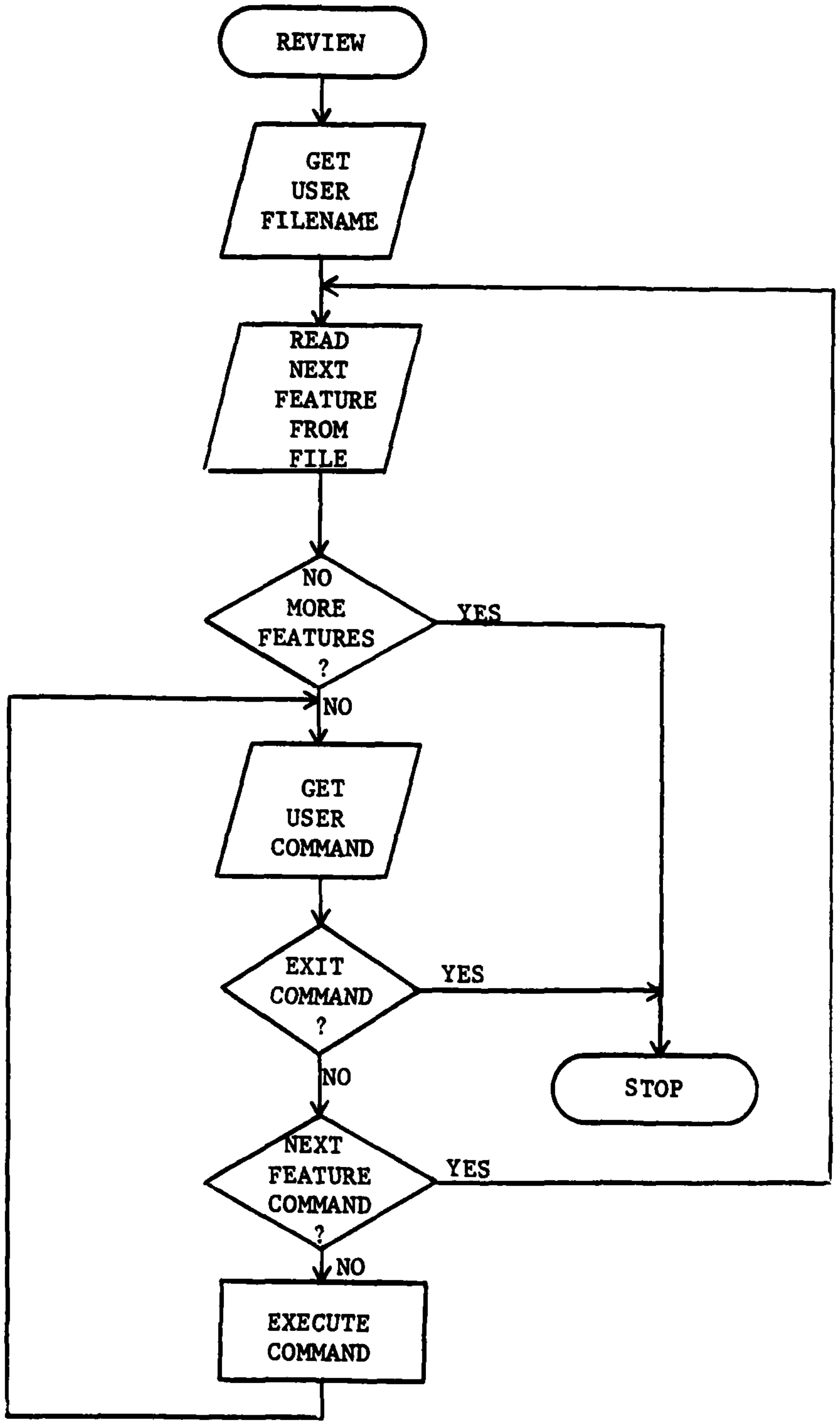
PROGRAM MANFILE



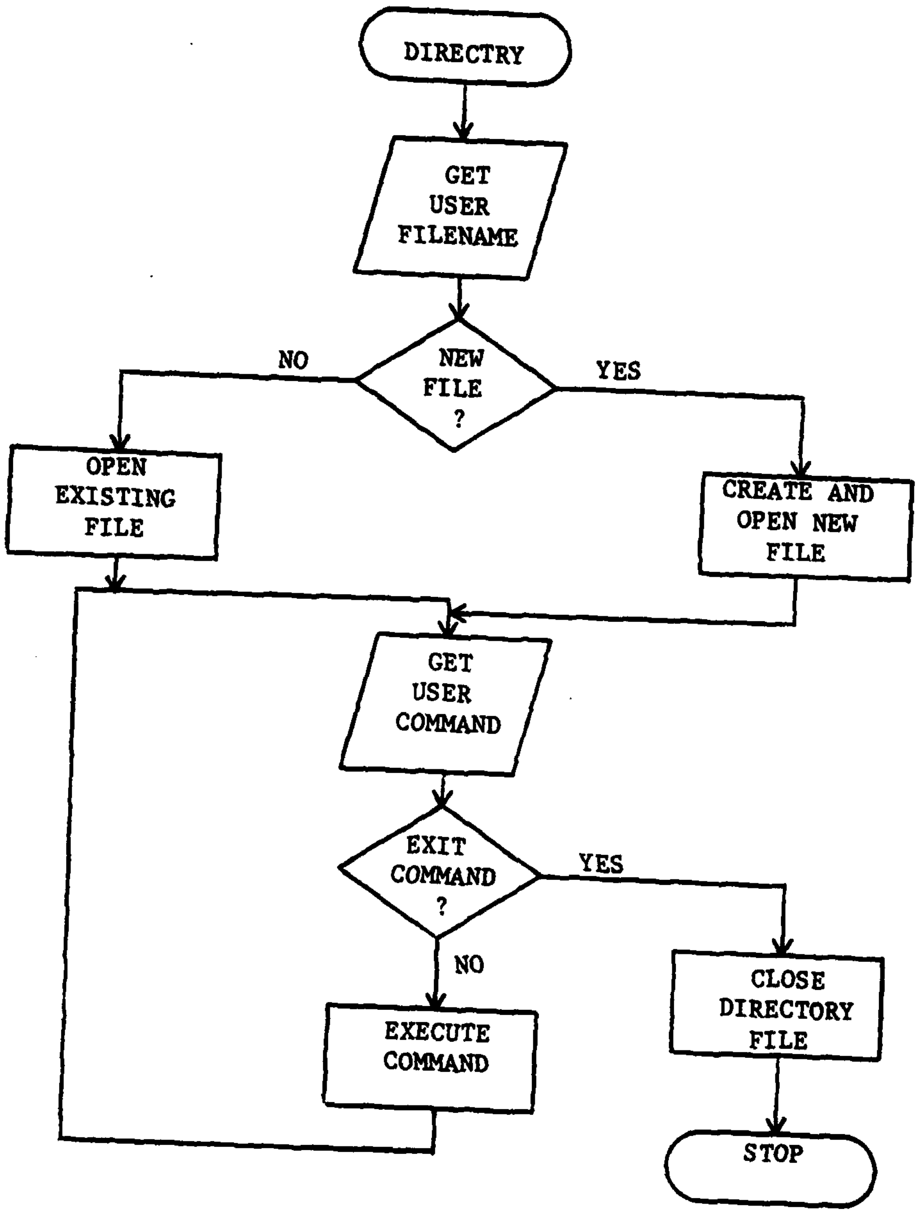
PROGRAM MANEDIT


MANEDIT
GET USER INPUT FILE, OUTPUT FILE
GET USER COMMAND
EXIT COMMAND
YES
NO
EXECUTE COMMAND
COPY REMAINING INPUT FILE DATA TO OUTPUT FILE
STOP

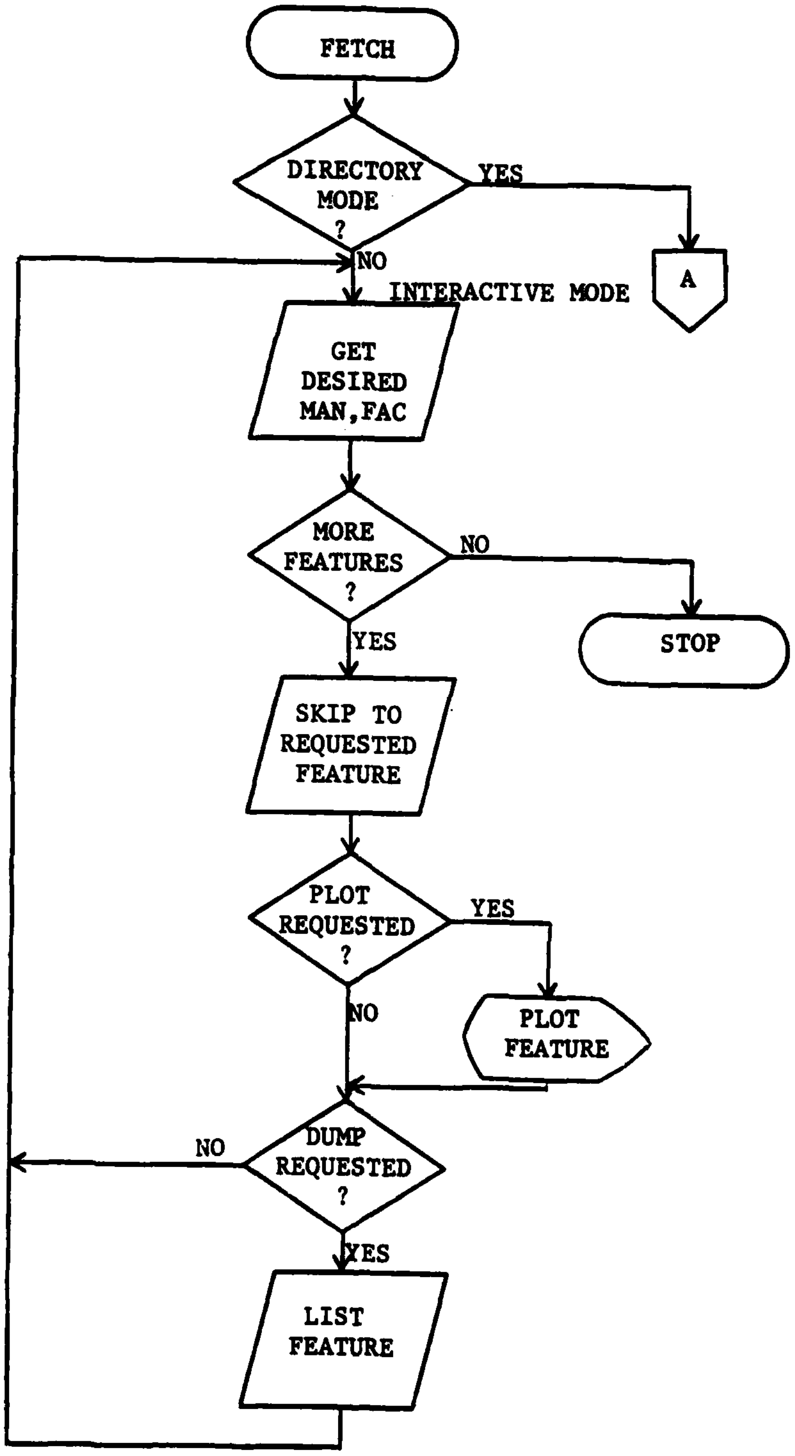
PROGRAM REVIEW



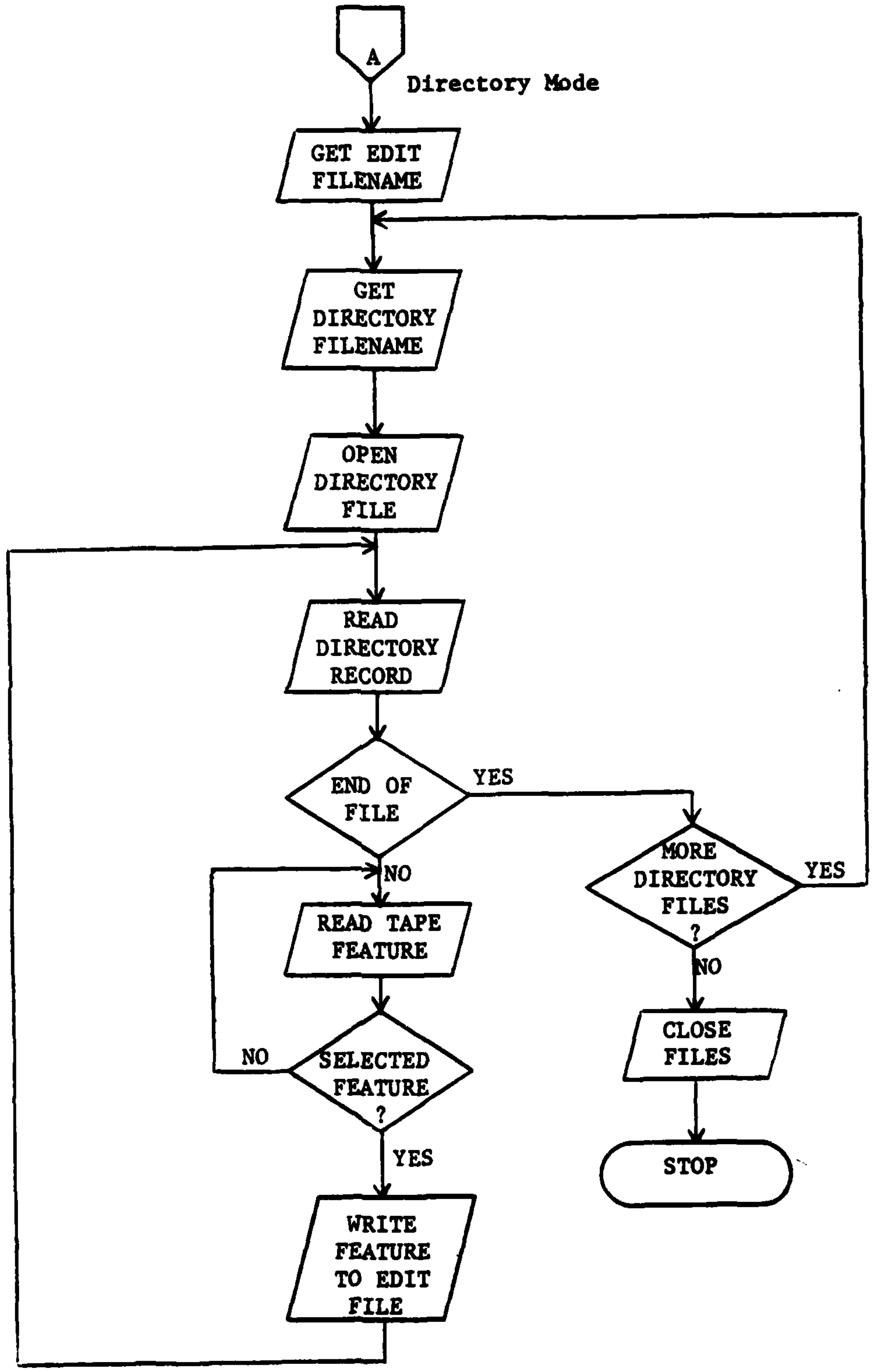
PROGRAM DIRECTRY



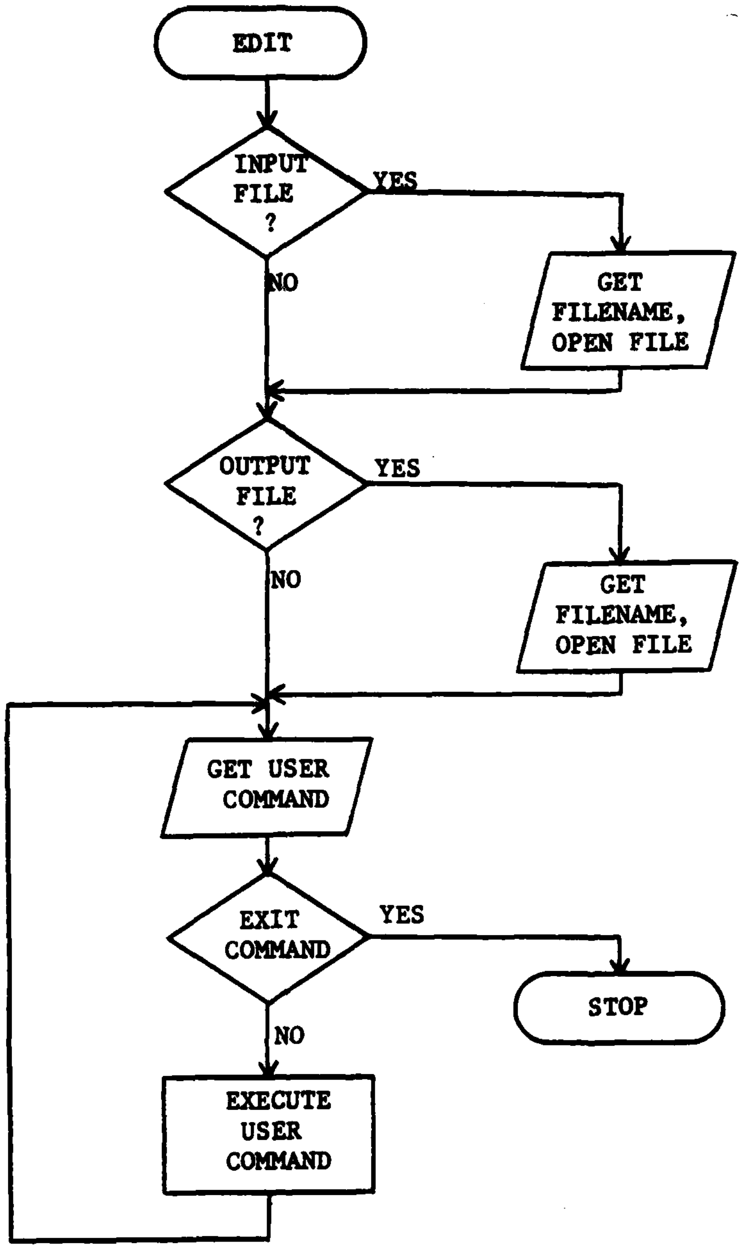
PROGRAM FETCH



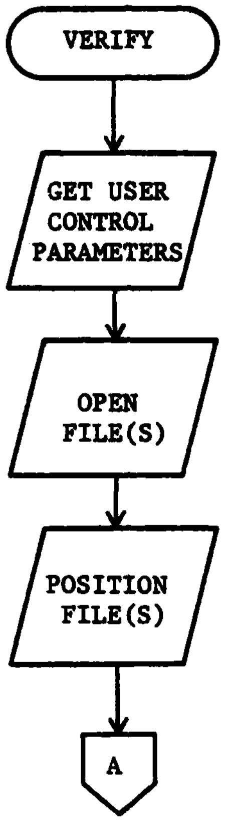
PROGRAM FETCH (cont'd)



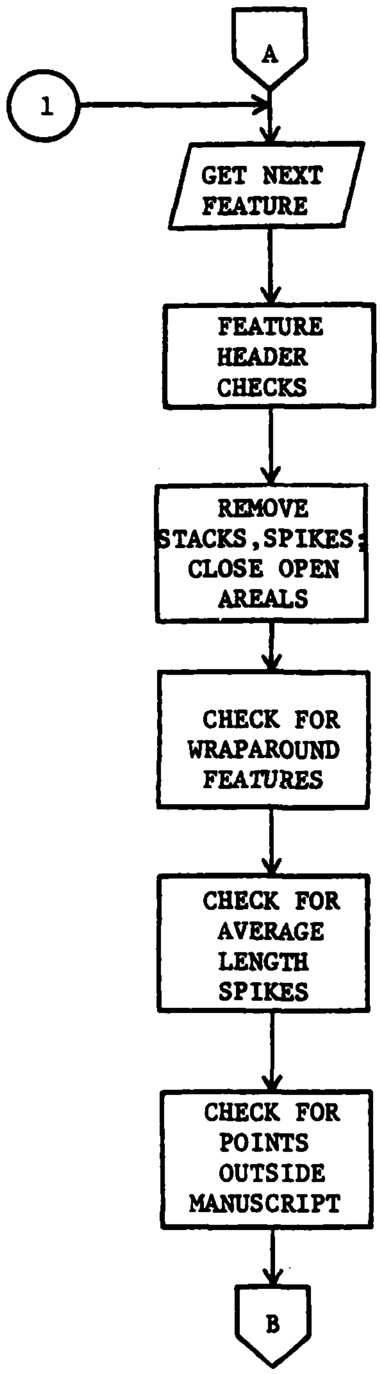
PROGRAM EDIT



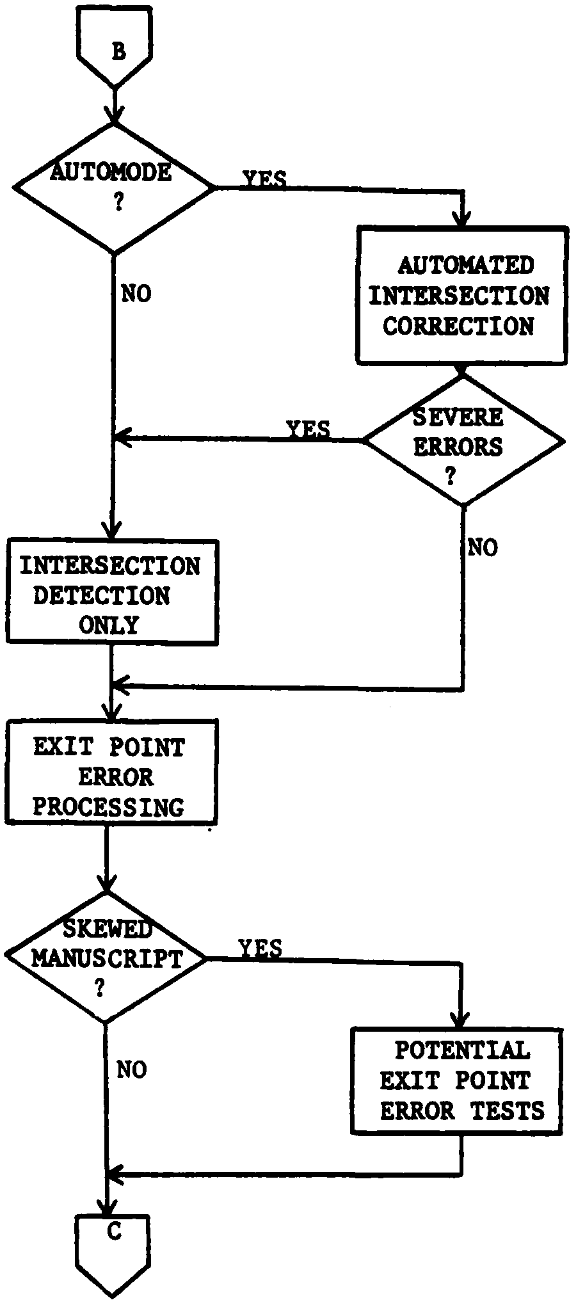
PROGRAM VERIFY



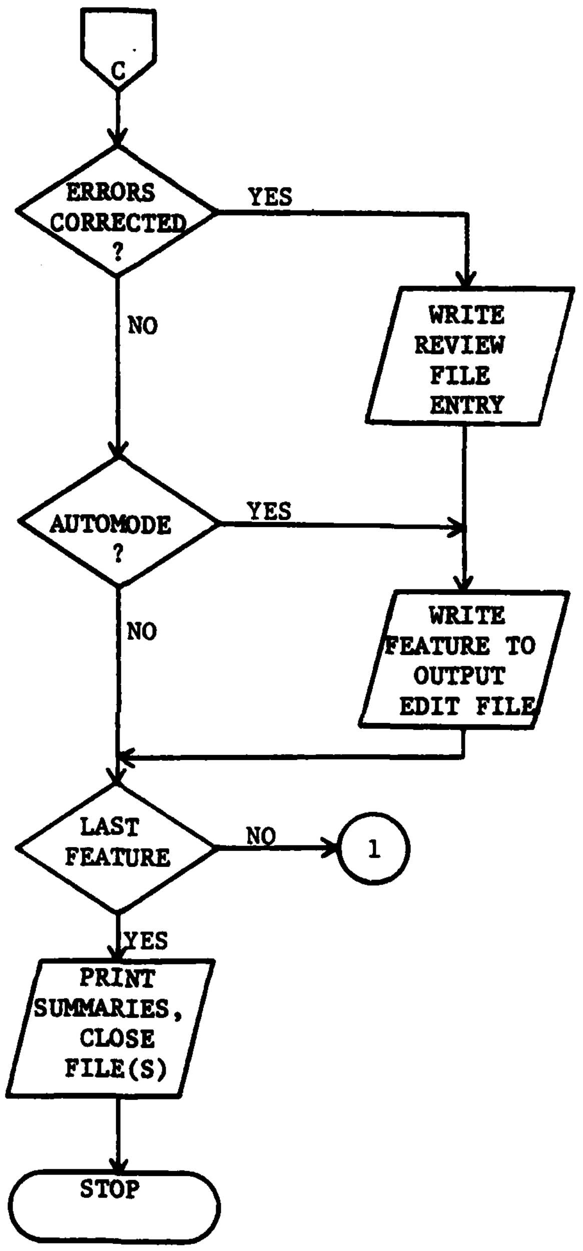
PROGRAM VERIFY (cont'd)


A
1
GET NEXT FEATURE
FEATURE HEADER CHECKS
REMOVE STACKS,SPIKES; CLOSE OPEN AREALS
CHECK FOR WRAPAROUND FEATURES
CHECK FOR AVERAGE LENGTH SPIKES
CHECK FOR POINTS OUTSIDE MANUSCRIPT
B

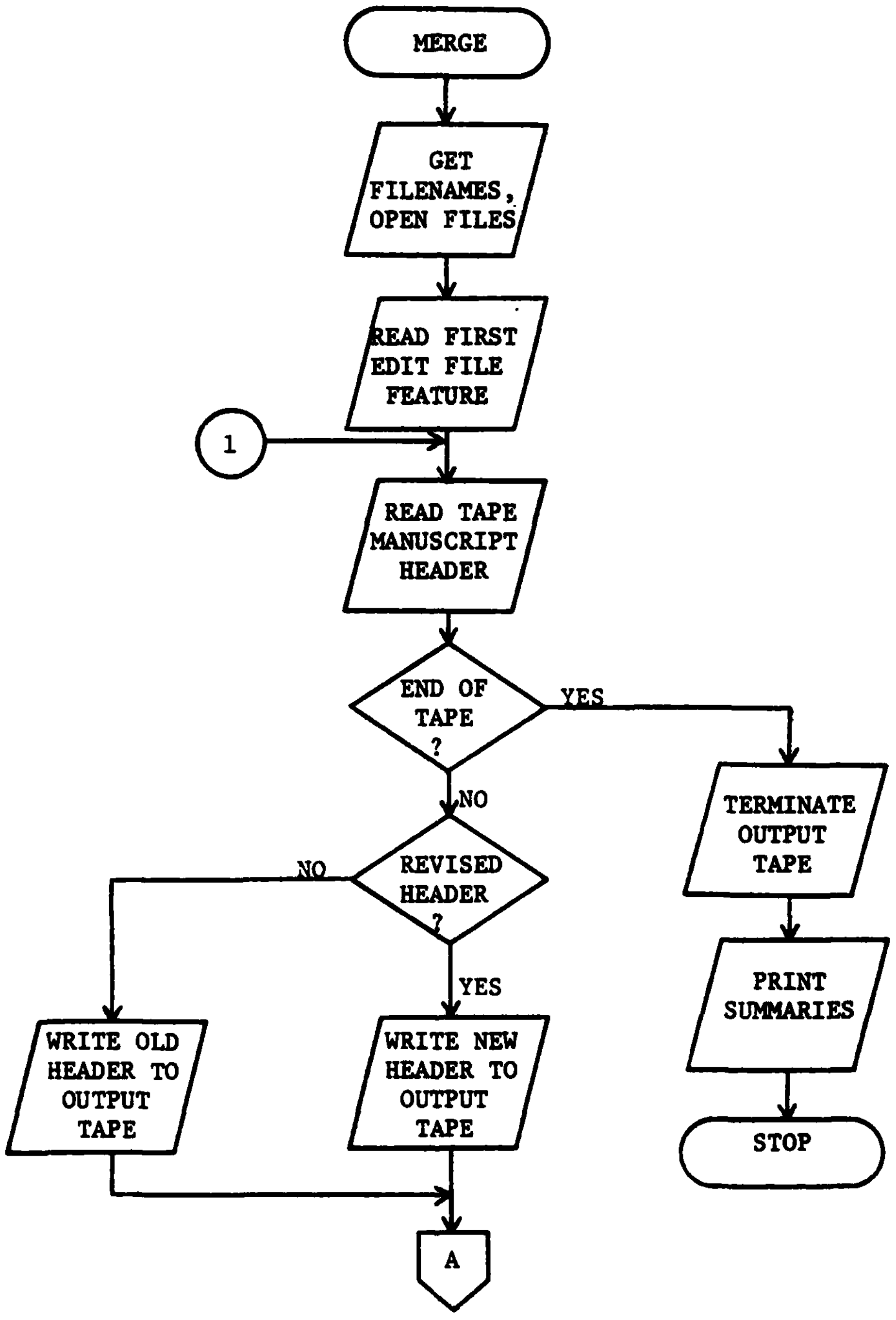
PROGRAM VERIFY (cont'd)


B
AUTOMODE ?
YES
NO
AUTOMATED INTERSECTION CORRECTION
SEVERE ERRORS ?
YES
NO
INTERSECTION DETECTION ONLY
EXIT POINT ERROR PROCESSING
SKEWED MANUSCRIPT ?
YES
NO
POTENTIAL EXIT POINT ERROR TESTS
C

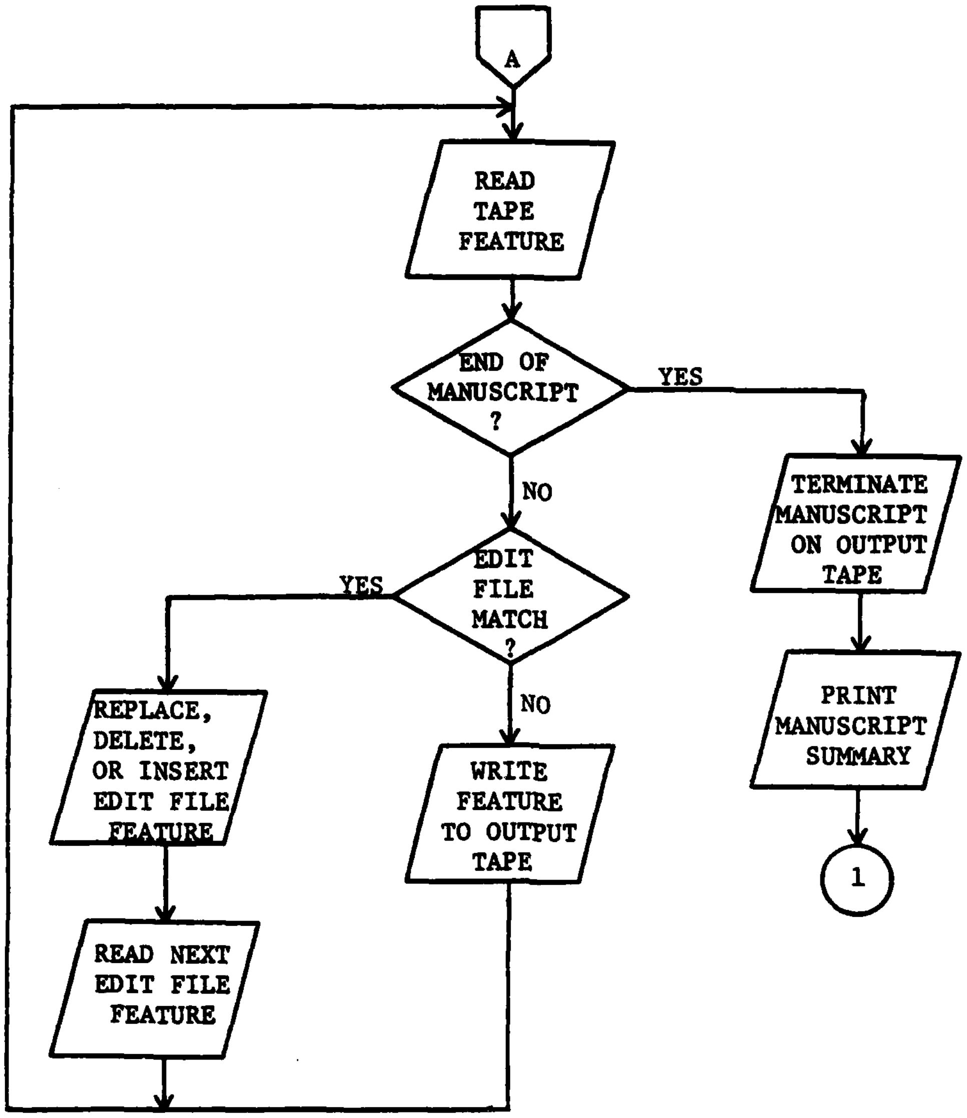
PROGRAM VERIFY (cont'd)



PROGRAM MERGE


MERGE
GET FILENAMES, OPEN FILES
READ FIRST EDIT FILE FEATURE
1
READ TAPE MANUSCRIPT HEADER
END OF TAPE ?
YES
NO
TERMINATE OUTPUT TAPE
PRINT SUMMARIES
STOP
REVISED HEADER ?
NO
YES
WRITE OLD HEADER TO OUTPUT TAPE
WRITE NEW HEADER TO OUTPUT TAPE
A

PROGRAM MERGE (cont'd)



A
READ TAPE FEATURE
END OF MANUSCRIPT ?
YES
NO
TERMINATE MANUSCRIPT ON OUTPUT TAPE
PRINT MANUSCRIPT SUMMARY
1
EDIT FILE MATCH ?
YES
NO
REPLACE, DELETE, OR INSERT EDIT FILE FEATURE
READ NEXT EDIT FILE FEATURE
WRITE FEATURE TO OUTPUT TAPE

VITA

Michael R. Nicol was born in Columbus, Ohio on July 23, 1953. He graduated from high school in Marysville, Ohio in 1971 and enrolled at Ohio Northern University. He graduated with a Bachelor of Science in Electrical Engineering in 1975. From 1975 to 1980, he was employed by the Air Force Human Resources Laboratory, where he was responsible for in-house and contractual development of sensor simulation techniques and data bases for training simulation. At the same time, he began work on a master's degree on a part time basis at the Air Force Institute of Technology. In 1980, he transferred to the Air Force Aeronautical System's Division Directorate of Equipment Engineering, Simulator Division, where he served as data base engineer for the visual system for the C-130 Weapon System Trainer. Since then his responsibilities have expanded to include development and evaluation of data base source material and data base generation techniques for simulator data bases.

Permanent Address: 5249 Woodcock Way
Dayton, Ohio 45424

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GCS/EE/82S-9	2. GOVT ACCESSION NO. AD-A124893	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) IMPROVING THE USEABILITY OF THE DEFENSE MAPPING AGENCY DIGITAL FEATURE ANALYSIS DATA FOR TRAINING SIMULATOR APPLICATIONS		5. TYPE OF REPORT & PERIOD COVERED MS Thesis 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Michael R. Nicol		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology Wright-Patterson AFB, Ohio 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Aeronautical Systems Division Visual and Electro Optical Branch (ASD/ENETV) Wright-Patterson AFB, Ohio 45433		12. REPORT DATE September 1982 13. NUMBER OF PAGES 214
14. MONITORING AGENCY NAME & ADDRESS(*if different from Controlling Office*)		15. SECURITY CLASS. (*of this report*) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (*of this Report*)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (*of the abstract entered in Block 20, if different from Report*)

18. SUPPLEMENTARY NOTES

[illegible] for public release: IAW AFR 190 17.

[illegible]
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (AIC)
Wright-Patterson AFB OH 45433

4 JAN 1983

19. KEY WORDS (*Continue on reverse side if necessary and identify by block number*)

Computer Image Generation (CIG)
Visual System
Simulation
Data Base
Interactive Computer Graphics
Digital Feature Analysis Data (DFAD)

20. ABSTRACT (*Continue on reverse side if necessary and identify by block number*)

The Air Force is intensifying its reliance upon flight simulators to provide initial and proficiency training for various aircrew members. The effectiveness of these simulators depends heavily upon the Defense Mapping Agency's digital cartographic data, which is used as source data to generate simulator data bases for visual, radar, and sensor simulations. Unfortunately, the source data occasionally displays certain characteristics which disrupt the highly automated processes used to generate these simulator data bases, or which lead

20. ABSTRACT

to improper or visually distracting effects in the resulting simulations.

This paper describes the development of a software package designed to detect these problems in the source data. The capability is also provided to interactively, or in some cases automatically, correct the problems, thus producing slightly modified data that is free of these disruptive characteristics. This software streamlines the simulator data base generation process and ultimately results in a higher fidelity simulator data base.

MEI

8